Cigales.java:

```
package cigales;


import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.ArrayList;

//L'he generat com a .jar i importat en lloc d'incloure-la com a classe del projecte perquè em detecti que és el mateix tipus que la parella emprada als projectes
//de coeficients de Butterworth (allà també importo la llibreria). Així, no requereix una conversió del tipus Parella dels projectes dels coeficients al tipus
//Parella d'aquest ni viceversa.
import Parella.Pair;
import java.io.FilenameFilter;
import java.io.InputStream;
import java.util.Arrays;
import java.util.Collections;

/**
   @brief Cigales és la classe que conté el codi principal de l'aplicació, tant pel que fa al processament del fitxer de so i l'anàlisi del cant com pel que fa a la
   * identificació de l'espècie de cigala a partir dels valors obtinguts durant aquesta anàlisi per a cada paràmetre tractat. Permet identificar l'espècie de cigala
   * a la qual correspon el cant enregistrat en un fitxer de so.
 */

public class Cigales {

   /** Indica l'ampliació que s'ha aplicat a les amplituds. */
   private static int decibelsIncrementats;

   /**
      @throws OutOfMemoryError
      @pre ---
      @post Retorna les dades d'àudio d'un fitxer de so.
      @return Les dades d'àudio d'un fitxer de so
      @param stream Flux de dades d'on s'extraurà la informació
   */
   private static DadesAudio obtinguesDadesAudio(InputStream stream) throws OutOfMemoryError, Exception {
      DadesAudio dadesAudio = new LectorWav().obtinguesFluxDAudio(stream);

      if( dadesAudio==null ) {
         throw new Exception("No s'ha pogut obtenir les dades d'àudio del fitxer.");
      } else {
         return dadesAudio;
      }
   }

   /**
      @throws OutOfMemoryError
      @pre ---
      @post Retorna les amplituds d'un fitxer de so. Tindrà tantes llistes com canals hi hagi al fitxer, i cada llista tindrà una mida corresponent als segons de duració del fitxer multiplicats per la freqüència de mostreig.
      @return Les amplituds d'un fitxer de so. Tindrà tantes llistes com canals hi hagi al fitxer, i cada llista tindrà una mida corresponent als segons de duració del fitxer multiplicats per la freqüència de mostreig.
      @param llistaBytes Llista de bytes del fitxer de so
      @param canals Nombre de canals de la gravació
   */
   private static int[][] obtinguesAmplitud(byte[] llistaBytes, int canals) throws OutOfMemoryError
   {
      int[][] llista = new int[canals][llistaBytes.length / (2 * canals)];
      int index = 0;

      for (int byteAudio = 0; byteAudio < llistaBytes.length;)
      {
         for (int canal = 0; canal < canals; canal++)
         {
            //Converteix el byte en una mostra d'amplitud.
            int baix = (int) llistaBytes[byteAudio];
            byteAudio++;
            int alt = (int) llistaBytes[byteAudio];
            byteAudio++;
            int mostra = (alt << 8) + (baix & 0x00ff);

            llista[canal][index] = mostra;
         }
         index++;
      }

      return llista;
   }

   /**
      @throws OutOfMemoryError
      @pre ---
      @post Retorna els valors d'amplitud entrats per paràmetre en forma de llista de bytes.
      @return Els valors d'amplitud entrats per paràmetre en forma de llista de bytes
      @param amplituds Llista d'amplituds del fitxer de so
      @param canals Nombre de canals de la gravació
   */
   private static byte[] obtinguesBytes (int[][] amplituds, int canals) throws OutOfMemoryError
   {
      byte[] llista = new byte[amplituds[0].length * 2 * canals];
      int index = 0;

      for (int valorAmplitud = 0; valorAmplitud < amplituds[0].length; valorAmplitud++)
      {
         for (int canal = 0; canal < canals; canal++)
         {
            if (amplituds[canal][valorAmplitud] > 0 || amplituds[canal][valorAmplitud] % 256 == 0) {
               llista[index+canal*2] = (byte) (amplituds[canal][valorAmplitud] % 256); //Baix
               llista[index+1+canal*2] = (byte) (amplituds[canal][valorAmplitud] / 256); //Alt
            }
            else {
               llista[index+canal*2] = (byte) (amplituds[canal][valorAmplitud] % 256); //Baix
               llista[index+1+canal*2] = (byte) (amplituds[canal][valorAmplitud]/256 - 1); //Alt
            }
         }
         index+=2*canals;
      }

      return llista;
   }

   /**
      @throws OutOfMemoryError
      @pre ---
      @post Retorna la llista de bytes entrada amb els valors tan amplificats com sigui possible sense que això provoqui una distorsió perceptible del so enregistrat.
      @return La llista de bytes entrada amb els valors tan amplificats com sigui possible sense que això provoqui una distorsió perceptible del so enregistrat
      @param llistaBytes Llista de bytes del fitxer de so
   */
   private static byte[] amplificaAmplituds(byte[] llistaBytes) throws OutOfMemoryError
   {
      byte[] llista = new byte[llistaBytes.length];

      //coefMultiplicacio = 10^(dB_Incrementats / 20)
      //dB_Incrementats = log_10(coefMultiplicacio)*20
      int dbIncrementats = 101;
      int coefMultiplicacio;

      int nValorsDistorsionats = llistaBytes.length;

      while (nValorsDistorsionats > llistaBytes.length*0.0001 && dbIncrementats > 0) {

         nValorsDistorsionats = 0;

         dbIncrementats--;
         coefMultiplicacio = (int)Math.round(Math.pow(10, dbIncrementats/20.0));

         for (int byteAudio = 0; byteAudio < llistaBytes.length - 1 && nValorsDistorsionats <= llistaBytes.length*0.0001; byteAudio += 2)
         {
            float mostra = (float)(llistaBytes[byteAudio] & 0xFF
                  | llistaBytes[byteAudio+1] << 8);

            mostra *= coefMultiplicacio;

            if ( mostra >= 32767f ) {
               nValorsDistorsionats++;
               llista[byteAudio] = (byte)0xFF;
               llista[byteAudio+1] = 0x7F;
            } else if ( mostra <= -32768f ) {
               nValorsDistorsionats++;
               llista[byteAudio] = 0x00;
               llista[byteAudio+1] = (byte)0x80;
            } else {
               int s = (int)( 0.5f + mostra );
               llista[byteAudio] = (byte)(s & 0xFF);
               llista[byteAudio+1] = (byte)(s >> 8 & 0xFF);
            }
         }
      }
```

```
    decibelsIncrementats = dbIncrementats;

    return llista;
  }

  /**
    @pre ---
    @post Troba la primera potència de p que sigui superior a x.
    @return La primera potència de p que sigui superior a x
    @param p Nombre del qual es vol obtenir una potència igual o superior a x
    @param x Nombre que hem d'igualar o superar amb una potència de p
  */
  private static int trobaPotenciaSuperiorOIgualA(int p, int x) {
    int y = 1;
    while (y < x) y *= p;
    return y;
  }

  /**
    @throws OutOfMemoryError
    @pre ---
    @post Converteix una ArrayList de reals a una llista de reals de mida equivalent a la primera potència de dos igual o superior a la mida de la llista original.
    @return Una llista de reals de mida equivalent a la primera potència de dos igual o superior a la mida de la llista original
    @param x ArrayList de reals
  */
  private static double[] deRealsARealsPerAFFT2(ArrayList<Integer> x) throws OutOfMemoryError {
    double[] llista = new double[trobaPotenciaSuperiorOIgualA(2, x.size())];
    int r = 0;
    while (r < x.size()) {
      llista[r] = x.get(r) * 1.0;
      r++;
    }

    //Amplia el vector perquè tingui una llargada que sigui potència de 2.
    while (r < llista.length) {
      llista[r] = 0.0;
      r++;
    }
    return llista;
  }

  /**
    @pre ---
    @post Redueix la llista x a p valors aplicant mitjanes
    @return La llista x escalada a p valors
    @param x ArrayList de reals que s'ha de reduir
    @param p Nombre de posicions que ha de tenir la llista resultant
  */
  private static ArrayList<Double> redueixA(ArrayList<Double> x, int p) {
    ArrayList<Double> llista = new ArrayList<>();

    int pas = (int) Math.round(x.size()*1.0/p);
    if (pas == 0) pas = 1;
    int r;
    int s;
    double auxiliar2;
    for(r = 0; r < x.size(); r+=pas) {
      auxiliar2 = 0;

      for(s = r; s < Math.min(r+pas, x.size()); s++) {
        auxiliar2 += x.get(s);
      }
      llista.add(1.0*auxiliar2/(s-r));
    }

    return llista;
  }

  /**
    @pre ---
    @post Combina les llistes x i y en una de sola assignant a cada posició la hipotenusa entre els valors de x i y per a aquesta mateixa posició, i escala la llista resultant a p posicions
    @return La combinació de les llistes x i y escalada a p valors
    @param x ArrayList de reals que s'ha de combinar i reduir
    @param y ArrayList de reals que s'ha de combinar i reduir
    @param p Nombre de posicions que ha de tenir la llista resultant
  */
  private static ArrayList<Double> realsRedueixA(double[] x, double[] y, int p) {
    ArrayList<Double> llista = new ArrayList<>();

    int pas = (int) Math.round(x.length*1.0/p);
    if (pas == 0) pas = 1;
    int r;
    int s;
    double auxiliar2;
    for(r = 0; r < x.length; r+=pas) {
      auxiliar2 = 0;
      for(s = r; s < Math.min(r+pas, x.length); s++) {
        auxiliar2 += Math.hypot(x[s], y[s]);
      }
      llista.add(1.0*auxiliar2/(s-r));
    }

    return llista;
  }

  /**
    @pre ---
    @post Retorna els coeficients de Butterworth filtrant per "band pass", és a dir, amb uns llindars mínim (x) i màxim (y) de la freqüència.
    @return Una parella que conté les llistes de coeficients de Butterworth obtinguts filtrant per "band pass", és a dir, amb uns llindars mínim (x) i màxim (y) de la freqüència
    @param x Llindar mínim de freqüència amb rang entre 0 i 1
    @param y Llindar màxim de freqüència amb rang entre 0 i 1
  */
  private static Pair<ArrayList<Double>, ArrayList<Double> > obtinguesCoeficientsBandPass(double x, double y) {

    if (x <= 0.03) return new Pair(coeficientsbutterworthbandpass.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass.Pair(x,y)).getSecond());
    else if (x <= 0.06) return new Pair(coeficientsbutterworthbandpass2.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass2.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass2.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass2.Pair(x,y)).getSecond());
    else if (x <= 0.095) return new Pair(coeficientsbutterworthbandpass3.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass3.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass3.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass3.Pair(x,y)).getSecond());
    else if (x <= 0.13) return new Pair(coeficientsbutterworthbandpass4.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass4.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass4.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass4.Pair(x,y)).getSecond());
    else if (x <= 0.165) return new Pair(coeficientsbutterworthbandpass5.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass5.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass5.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass5.Pair(x,y)).getSecond());
    else if (x <= 0.20) return new Pair(coeficientsbutterworthbandpass6.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass6.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass6.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass6.Pair(x,y)).getSecond());
    else if (x <= 0.24) return new Pair(coeficientsbutterworthbandpass7.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass7.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass7.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass7.Pair(x,y)).getSecond());
    else if (x <= 0.28) return new Pair(coeficientsbutterworthbandpass8.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass8.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass8.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass8.Pair(x,y)).getSecond());
    else if (x <= 0.325) return new Pair(coeficientsbutterworthbandpass9.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass9.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass9.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass9.Pair(x,y)).getSecond());
    else if (x <= 0.375) return new Pair(coeficientsbutterworthbandpass10.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass10.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass10.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass10.Pair(x,y)).getSecond());
    else if (x <= 0.425) return new Pair(coeficientsbutterworthbandpass11.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass11.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass11.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass11.Pair(x,y)).getSecond());
    else if (x <= 0.485) return new Pair(coeficientsbutterworthbandpass12.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass12.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass12.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass12.Pair(x,y)).getSecond());
    else if (x <= 0.55) return new Pair(coeficientsbutterworthbandpass13.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass13.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass13.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass13.Pair(x,y)).getSecond());
```

```
    else if (x <= 0.63) return new Pair(coeficientsbutterworthbandpass14.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass14.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass14.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass14.Pair(x,y)).getSecond());
    else if (x <= 0.73) return new Pair(coeficientsbutterworthbandpass15.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass15.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass15.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass15.Pair(x,y)).getSecond());
    else if (x <= 0.90) return new Pair(coeficientsbutterworthbandpass16.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass16.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass16.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass16.Pair(x,y)).getSecond());
    else return new Pair(coeficientsbutterworthbandpass17.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass17.Pair(x,y)).getFirst(), coeficientsbutterworthbandpass17.AssignaCoeficients.assignaCoeficientsButterworthFiltrePassBand().get(new coeficientsbutterworthbandpass17.Pair(x,y)).getSecond());
  }

  /**
    @pre ---
    @post Retorna l'arrodoniment de x a una precisió de 1/y.
    @return L'arrodoniment de x a una precisió de 1/y
    @param x Valor que s'ha d'arrodonir
    @param y Valor que indica la precisió de l'arrodoniment
  */
  private static double arrodoneixADivisio(double x, int y) {
    if (x % (1.0/y) >= 1.0/(2*y)) return Math.round(1000.0 * (x - x % (1.0/y) + 1.0/y))/1000.0;
    else return Math.round(1000.0*(x - x % (1.0/y)))/1000.0;
  }

  /**
    @pre ---
    @post Troba la mitjana dels primers durada elements de la llista partint de la posició index
    @return La mitjana dels primers durada elements de la llista partint de la posició index
    @param index Posició de la llista en què comencem a fer la mitjana
    @param durada Nombre de posicions de les quals hem de fer la mitjana
    @param llista Llista d'on extraiem els valors per fer la mitjana
  */
  private static double trobaMitjana(int index, int durada, ArrayList<Double> llista) {
    if (durada == 0 || llista.isEmpty()) return 0.0;
    else {
      double mitjana = 0.0;
      int nIteracions = 0;
      while (nIteracions < durada && nIteracions+index < llista.size()) {
        mitjana += llista.get(nIteracions+index);
        nIteracions++;
      }
      if (nIteracions == 0) return 0.0;
      else return mitjana / nIteracions;
    }
  }

  /**
    @pre ---
    @post Troba la mitjana dels primers durada elements de la llista partint de la posició index, però ponderant d'una manera superior el valor màxim dins d'aquestes posicions
    @return La mitjana dels primers durada elements de la llista partint de la posició index, però ponderant d'una manera superior el valor màxim dins d'aquestes posicions
    @param index Posició de la llista en què comencem a fer la mitjana
    @param durada Nombre de posicions de les quals hem de fer la mitjana
    @param llista Llista d'on extraiem els valors per fer la mitjana
    @param posicionsValidesMaxim Indica per a quines posicions ponderem el màxim de manera superior als alters valors. Si és un número negatiu, només ponderarà de manera especial el màxim a les últimes posicionsValidesMaxim posicions, i si és un número positiu només ho faré a les primeres posicionsValidesMaxim posicions.
  */
  private static double trobaMitjanaAmbMaxims(int index, int durada, ArrayList<Double> llista, int posicionsValidesMaxim) {
    //Si entren un número negatiu a posicionsValidesMaxim, només mirarà el màxim a les últimes posicionsValidesMaxim posicions.
    //En canvi, si entren un número positiu, només mirarà el màxim a les primeres posicionsValidesMaxim posicions.
    if (durada == 0 || llista.isEmpty()) return 0.0;
    else {
      double maxim = 0.0;
      double mitjana = 0.0;
      int nIteracions = 0;
      while (nIteracions < durada && nIteracions+index < llista.size()) {
        mitjana += llista.get(nIteracions+index);
        if (llista.get(nIteracions+index) > maxim && ((posicionsValidesMaxim > 0 && nIteracions <= posicionsValidesMaxim) || (posicionsValidesMaxim < 0 && durada - nIteracions <= posicionsValidesMaxim))) maxim = llista.get(nIteracions+index);
        nIteracions++;
      }
      return ((mitjana / nIteracions)*3.75 + maxim*1.25) / 5.0;
    }
  }

  /**
    @pre ---
    @post Retorna una parella que inclou la mitjana dels primers durada elements de la llista partint de la posició index, ponderant d'una manera superior el valor màxim dins d'aquestes posicions, i aquest valor màxim.
    @return Una parella que inclou la mitjana dels primers durada elements de la llista partint de la posició index, ponderant d'una manera superior el valor màxim dins d'aquestes posicions, i aquest valor màxim
    @param index Posició de la llista en què comencem a fer la mitjana
    @param durada Nombre de posicions de les quals hem de fer la mitjana
    @param llista Llista d'on extraiem els valors per fer la mitjana
    @param posicionsValidesMaxim Indica per a quines posicions ponderem el màxim de manera superior als alters valors. Si és un número negatiu, només ponderarà de manera especial el màxim a les últimes posicionsValidesMaxim posicions, i si és un número positiu només ho faré a les primeres posicionsValidesMaxim posicions.
  */
  private static Pair<Double, Double> trobaParellaMitjanaMaxim(int index, int durada, ArrayList<Double> llista, int posicionsValidesMaxim) {
    //Si entren un número negatiu a posicionsValidesMaxim, només mirarà el màxim a les últimes posicionsValidesMaxim posicions.
    //En canvi, si entren un número positiu, només mirarà el màxim a les primeres posicionsValidesMaxim posicions.
    if (durada == 0 || llista.isEmpty()) return new Pair(0.0, 0.0);
    else {
      double maxim = 0.0;
      double mitjana = 0.0;
      int nIteracions = 0;
      while (nIteracions < durada && nIteracions+index < llista.size()) {
        mitjana += llista.get(nIteracions+index);
        if (llista.get(nIteracions+index) > maxim && ((posicionsValidesMaxim > 0 && nIteracions <= posicionsValidesMaxim) || (posicionsValidesMaxim < 0 && durada - nIteracions <= posicionsValidesMaxim))) maxim = llista.get(nIteracions+index);
        nIteracions++;
      }
      return new Pair(((mitjana / nIteracions)*3.75 + maxim*1.25) / 5.0, maxim);
    }
  }

  /**
    @pre ---
    @post Retorna una parella que inclou la mitjana i el màxim entrats actualitzats de la manera que ens indiquen els paràmetres entrats.
    @return Una parella que inclou la mitjana i el màxim entrats actualitzats de la manera que ens indiquen els paràmetres entrats.
    @param index Posició de la llista en què ha de començar la mitjana
    @param durada Nombre de posicions de les quals hem de fer la mitjana (si nomesFinal és cert, hem de fer la mitjana amb pas posicions més)
    @param llista Llista d'on extraiem els valors per fer la mitjana
    @param posicionsValidesMaxim Indica per a quines posicions ponderem el màxim de manera superior als alters valors. Si és un número negatiu, només ponderaré de manera especial el màxim a les últimes posicionsValidesMaxim posicions, i si és un número positiu només ho faré a les primeres posicionsValidesMaxim posicions.
    @param mitjanaAnterior Mitjana que hem d'actualitzar
    @param maximAnterior Màxim que hem d'actualitzar
    @param pas Nombre de posicions que hem d'avançar els anteriors màxim i mitjana
    @param nomesFinal Booleà que ens indica, en cas que sigui cert, que no hem de desplaçar la mitjana sinó ampliar-la pas posicions en cas que l'ampliem pel final o -pas posicions en cas que l'ampliem per l'inici
    @param duradaInferior Booleà que ens indica si la nova mitjana ha d'incloure menys valors que l'anterior
  */
  private static Pair<Double, Double> actualitzaMitjanaAmbMaxims(int index, int durada, ArrayList<Double> llista, int posicionsValidesMaxim, double mitjanaAnterior, double maximAnterior, int pas, boolean nomesFinal, boolean duradaInferior) {
    //Si entren un número negatiu a posicionsValidesMaxim, només mirarà el màxim a les últimes posicionsValidesMaxim posicions.
    //En canvi, si entren un número positiu, només mirarà el màxim a les primeres posicionsValidesMaxim posicions.
    if (durada == 0 || llista.isEmpty() || index-1+durada >= llista.size()) return new Pair(0.0, 0.0);
    else if (maximAnterior == 0.0 || index == 0 || maximAnterior == llista.get(index-1)|| mitjanaAnterior == 0.0) return trobaParellaMitjanaMaxim(index, durada, llista, posicionsValidesMaxim);
    else if (nomesFinal) {
      double mitjana = mitjanaAnterior;
      double maxim;
      if (posicionsValidesMaxim > 0 && index-1+posicionsValidesMaxim < llista.size() && index-1+posicionsValidesMaxim >= 0)
        maxim = Math.max(llista.get(index-1+posicionsValidesMaxim), maximAnterior);
```

```
        else if (posicionsValidesMaxim < 0 && index-1+durada-posicionsValidesMaxim < llista.size() && index-1+durada-posicionsValidesMaxim >= 0)
            maxim = Math.max(llista.get(index-1+durada-posicionsValidesMaxim), maximAnterior);
        else maxim = maximAnterior;
        mitjana += (maxim - maximAnterior)/5.0;
        mitjana = ((mitjana * (durada-1)) + ((llista.get(index-1+durada)*3.75 + maxim*1.25)/5.0)) / durada;
        return new Pair(mitjana, maxim);
      }
      else {
        //Avança les posicions a partir de les quals fa la mitjana.
        if (pas == 1) {
          double mitjana = mitjanaAnterior;
          double maxim;
          if (posicionsValidesMaxim > 0 && index-1+posicionsValidesMaxim < llista.size() && index-1+posicionsValidesMaxim >= 0)
            maxim = Math.max(llista.get(index-1+posicionsValidesMaxim), maximAnterior);
          else if (posicionsValidesMaxim < 0 && index-1+durada-posicionsValidesMaxim < llista.size() && index-1+durada-posicionsValidesMaxim >= 0)
            maxim = Math.max(llista.get(index-1+durada-posicionsValidesMaxim), maximAnterior);
          else maxim = maximAnterior;
          if (!duradaInferior) {
            mitjana += (maxim - maximAnterior)/5.0;
            mitjana += ((llista.get(index-1+durada)*3.75 + maxim*1.25)/5.0 - (llista.get(index-1)*3.75 + maxim*1.25)/5.0) / durada;
          }
          else { //Ha avançat una posició (l'inici) a la llista, però manté el final, és a dir, agrupa un valor d'amplitud menys que abans.
            mitjana = (mitjana * (durada+1) - (llista.get(index-1)*3.75 + maximAnterior*1.25)/5.0) / durada;
            mitjana += (maxim - maximAnterior)/5.0;
          }
          return new Pair(mitjana, maxim);
        }
        //Retrocedeix les posicions a partir de les quals fa la mitjana.
        else if (pas == -1) {
          double mitjana = mitjanaAnterior;
          double maxim = Math.max(llista.get(index), maximAnterior);
          if (!duradaInferior) {
            mitjana += (maxim - maximAnterior)/5.0;
            mitjana += ((llista.get(index)*3.75 + maxim*1.25)/5.0 - (llista.get(index+durada)*3.75 + maxim*1.25)/5.0) / durada;
          }
          else { //Ha retrocedit una posició (el final) a la llista, però manté el principi, és a dir, agrupa un valor d'amplitud menys que abans.
            mitjana = (mitjana * (durada+1) - (llista.get(index+durada)*3.75 + maximAnterior*1.25)/5.0) / durada;
            mitjana += (maxim - maximAnterior)/5.0;
          }
          return new Pair(mitjana, maxim);
        }
        //No ha de passar.
        else return new Pair(0.0, 0.0);
      }
    }

    /**
      @pre ---
      @post Retorna la mitjana entrada actualitzada de la manera que ens indiquen els paràmetres entrats.
      @return La mitjana entrada actualitzada de la manera que ens indiquen els paràmetres entrats
      @param inici Posició de la llista en què ha de començar la mitjana
      @param fi Posició de la llista en què he d'acabar la mitjana
      @param llista Llista d'on extraiem els valors per fer la mitjana
      @param mitjanaAnterior Mitjana que hem d'actualitzar
      @param nomesFinal Booleà que ens indica, en cas que sigui cert, que no hem de desplaçar la mitjana sinó ampliar-la una posició (la següent a l'última tinguda en compte per a l'anterior mitjana)
    */
    private static double actualitzaMitjanaLocalAmbValorsAbsolutsPas1(int inici, int fi, ArrayList<Double> llista, double mitjanaAnterior, boolean nomesFinal) {
      if (inici < 0 || fi >= llista.size() || fi-inici+1 <= 0) return 0.0;
      else if (inici == 0 || mitjanaAnterior == 0.0) return trobaMitjanaLocalAmbValorsAbsoluts(inici, fi, llista);
      else if (!nomesFinal) {
        return mitjanaAnterior + (Math.abs(llista.get(fi)) - Math.abs(llista.get(inici-1))) / (fi-inici+1);
      }
      else { //Només avança el final, no el principi.
        return ((mitjanaAnterior * (fi-inici)) + llista.get(fi)) / (fi-inici+1);
      }
    }

    /**
      @pre ---
      @post Troba la mitjana dels primers durada elements de la llista en valor absolut partint de la posició index
      @return La mitjana dels primers durada elements de la llista en valor absolut partint de la posició index
      @param index Posició de la llista en què comencem a fer la mitjana
      @param durada Nombre de posicions de les quals hem de fer la mitjana
      @param llista Llista d'on extraiem els valors per fer la mitjana
    */
    private static double trobaMitjanaAmbValorsAbsoluts(int index, int durada, ArrayList<Double> llista) {
      double mitjana = 0.0;
      int nIteracions = 0;
      while (nIteracions < durada && nIteracions+index < llista.size()) {
        mitjana += Math.abs(llista.get(nIteracions+index));
        nIteracions++;
      }
      return mitjana / nIteracions;
    }

    /**
      @pre ---
      @post Troba la freqüència de mostreig real de l'enregistrament a partir del qual hem obtingut la transformada de Fourier els valors de la qual conté la llista llistaFreq
      @return La freqüència de mostreig real de l'enregistrament a partir del qual hem obtingut la transformada de Fourier els valors de la qual conté la llista llistaFreq
      @param llistaFreq Llista de valors de la transformada de Fourier d'un fitxer de so
      @param fs Freqüència de mostreig teòrica, que no sempre és la real
    */
    private static Integer trobaFsReal(ArrayList<Double> llistaFreq, int fs) {
      int index = llistaFreq.size()/2;
      while (index < llistaFreq.size() && !(trobaMitjana(index, llistaFreq.size()-index - 1, llistaFreq) < trobaMitjana(0, index, llistaFreq) * 0.005)) {
        index++;
      }
      return (int) Math.round(((fs/2.0)*index/llistaFreq.size())/1000)*1000;
    }

    /**
      @pre ---
      @post Troba tres pics d'amplitud respecte a la freqüència que puguin correspondre a una cigala del gènere Tibicina a partir dels valors de la llista llistaFreq
      @return Una llista amb les posicions dels tres pics d'amplitud respecte a la freqüència que puguin correspondre a una cigala del gènere Tibicina a partir dels valors de la llista llistaFreq. En cas que no s'hagi trobat tres pics que compleixin els requeriments, es retorna simplement la posició del pic d'amplitud més alt.
      @param llistaFreq Llista de valors de la transformada de Fourier d'un fitxer de so
      @param fs Freqüència de mostreig del fitxer de so
    */
    private static ArrayList<Integer> trobaTresPics(ArrayList<Double> llistaFreq, int fs) {
      ArrayList<Integer> pics = new ArrayList<>();
      double augmentFreq = 0.5*fs/llistaFreq.size();

      //Afegeix tots els pics possibles
      for(int i = 3; i < llistaFreq.size()-3; i++) {
        if (llistaFreq.get(i) > llistaFreq.get(i-1) && llistaFreq.get(i) > llistaFreq.get(i+1)) {
          pics.add(i);
        }
      }

      //Elimina els pics que no puguin correspondre a una cigala del gènere Tibicina
      for(int i = 0; i < pics.size(); i++) {
        if (pics.get(i)*augmentFreq < 5000 || pics.get(i)*augmentFreq > 13000) {
          pics.remove(i);
          i--;
        }
      }

      ArrayList<ArrayList<Integer> > llistaTresPics = new ArrayList<>();

      int maxim = 0;
      for(int i = 1; i < pics.size(); i++) {
        if (llistaFreq.get(pics.get(i)) > llistaFreq.get(pics.get(maxim))) maxim = i;
      }

      if (pics.size() >= 3) {
        ArrayList<Integer> aux;
        for(int i = 0; i < pics.size()-2; i++) {
          for(int j = i+1; j < pics.size()-1; j++) {
            for(int k = j+1; k < pics.size(); k++) {
              //Els tres pics no poden estar excessivament separats i el pic 2 ha de ser més alt que el pic 1 i que el pic 3.
              if (pics.get(j) - pics.get(i) < 3500/augmentFreq && pics.get(k) - pics.get(j) < 3500/augmentFreq && pics.get(k) - pics.get(i) < 6000/augmentFreq &&
                  pics.get(j) - pics.get(i) > 600/augmentFreq && pics.get(k) - pics.get(j) > 600/augmentFreq && pics.get(k) - pics.get(i) > 1400/augmentFreq &&
                  llistaFreq.get(pics.get(i)) < llistaFreq.get(pics.get(j))*1.5 && llistaFreq.get(pics.get(k)) < llistaFreq.get(pics.get(j))*1.2
                  ) {
                aux = new ArrayList<>();
                aux.add(pics.get(i));
                aux.add(pics.get(j));
                aux.add(pics.get(k));
                llistaTresPics.add(aux);
```

```
            }
          }
        }
      }
    }

    //Si només ha trobat un trio de pics, retorna aquest.
    if (llistaTresPics.size() == 1) {
      for(int i = 0; i < llistaTresPics.get(0).size(); i++) llistaTresPics.get(0).set(i, (int) Math.round(llistaTresPics.get(0).get(i)*augmentFreq));
      return llistaTresPics.get(0);
    }
    //Si ha trobat més d'un trio de pics, retorna el que sumi uns valors d'amplitud més alts en total.
    else if (llistaTresPics.size() > 1) {
      int trioSeleccionat = -1;
      for(int i = 0; i < llistaTresPics.size(); i++) {
        if (trioSeleccionat == -1 || llistaFreq.get(llistaTresPics.get(trioSeleccionat).get(0)) + llistaFreq.get(llistaTresPics.get(trioSeleccionat).get(1)) + llistaFreq.get(llistaTresPics.get(trioSeleccionat).get(2)) < llistaFreq.get(llistaTresPics.get(i).get(0)) + llistaFreq.get(llistaTresPics.get(i).get(1)) + llistaFreq.get(llistaTresPics.get(i).get(2))) trioSeleccionat = i;
      }
      for(int i = 0; i < llistaTresPics.get(trioSeleccionat).size(); i++) llistaTresPics.get(trioSeleccionat).set(i, (int) Math.round(llistaTresPics.get(trioSeleccionat).get(i)*augmentFreq));
      return llistaTresPics.get(trioSeleccionat);
    }
    //Si no ha trobat cap trio de pics, retorna simplement el valor més alt.
    else {
      ArrayList<Integer> picMesAlt = new ArrayList<>();
      picMesAlt.add((int) Math.round(pics.get(maxim)*augmentFreq));
      return picMesAlt;
    }
  }

  /**
    @pre ---
    @post Retorna una llista equivalent a la llista llista però amb el valor absolut de tots els seus valors
    @return Una llista equivalent a la llista llista però amb el valor absolut de tots els seus valors
    @param llista Llista de reals
  */
  private static ArrayList<Double> passaAValorAbsolut(ArrayList<Double> llista) {
    ArrayList<Double> resultat = new ArrayList<>();
    for (Double llista1 : llista) {
      resultat.add(Math.abs(llista1));
    }
    return resultat;
  }

  /**
    @pre ---
    @post Retorna el valor màxim de la llista llista que sigui inferior al valor limit.
    @return El valor màxim de la llista llista que sigui inferior al valor limit
    @param llista Llista de reals
    @param limit Real que indica un valor que ha de ser superior al màxim retornat
  */
  private static double trobaMaximInferiorA(ArrayList<Double> llista, double limit) {
    double maxim = 0;
    for (Double llista1 : llista) {
      if (llista1 > maxim && llista1 < limit) {
        maxim = llista1;
      }
    }
    return maxim;
  }

  /**
    @pre ---
    @post Retorna el valor màxim de la llista llista entre les posicions inici i fi.
    @return El valor màxim de la llista llista entre les posicions inici i fi
    @param llista Llista de reals
    @param inici Posició a partir de la qual comencem a tenir en compte els valors per trobar el màxim
    @param fi Posició a partir de la qual deixem de tenir en compte els valors per trobar el màxim
  */
  private static double trobaMaximLocal(ArrayList<Double> llista, int inici, int fi) {
    double maxim = 0;
    for(int i = inici; i <= fi; i++) {
      if (llista.get(i) > maxim) maxim = llista.get(i);
    }
    return maxim;
  }

  /**
    @pre ---
    @post Retorna el valor mínim de la llista entrada.
    @return El valor mínim de la llista entrada
    @param llista Llista de reals
  */
  private static double trobaMinim(ArrayList<Double> llista) {
    double minim = Double.MAX_VALUE;
    for (Double llista1 : llista) {
      if (llista1 < minim) {
        minim = llista1;
      }
    }
    return minim;
  }

  /**
    @pre ---
    @post Retorna el valor mínim de la llista llista entre les posicions inici i fi.
    @return El valor mínim de la llista llista entre les posicions inici i fi
    @param llista Llista de reals
    @param inici Posició a partir de la qual comencem a tenir en compte els valors per trobar el mínim
    @param fi Posició a partir de la qual deixem de tenir en compte els valors per trobar el mínim
  */
  private static double trobaMinimLocal(ArrayList<Double> llista, int inici, int fi) {
    double minim = Double.MAX_VALUE;
    for(int i = inici; i <= fi; i++) {
      if (llista.get(i) < minim) minim = llista.get(i);
    }
    return minim;
  }

  /**
    @pre ---
    @post Troba la mitjana dels elements de la llista compresos entre les posicions inici i fi en valor absolut
    @return La mitjana dels elements de la llista compresos entre les posicions inici i fi en valor absolut
    @param inici Posició de la llista en què comencem a fer la mitjana
    @param fi Posició de la llista en què acabem la mitjana
    @param llista Llista d'on extraiem els valors per fer la mitjana
  */
  private static double trobaMitjanaLocalAmbValorsAbsoluts(int inici, int fi, ArrayList<Double> llista) {
    double mitjana = 0;
    for(int i = inici; i <= fi; i++) {
      mitjana += Math.abs(llista.get(i));
    }
    return mitjana / (fi-inici+1);
  }

  /**
    @pre ---
    @post Retorna la llista de cants entrada però havent-los ajustat l'inici en cas que aquest definís un cant més llarg del que toca a causa de la manca de precisió.
    @return La llista de cants entrada però havent-los ajustat l'inici en cas que aquest definís un cant més llarg del que toca a causa de la manca de precisió.
    @param llistaCants Llista dels cants l'inici dels quals s'ha d'intentar precisar millor
    @param amplitudsModificadesAbsolutesReduides Llista d'amplituds filtrades, en valor absolut i escalades a un nombre de valors més petit que permetran determinar amb més precisió l'inici dels cants entrats
  */
  private static ArrayList<Pair<Double, Double> > ajustaCants(ArrayList<Pair<Double, Double> > llistaCants, ArrayList<Double> amplitudsModificadesAbsolutesReduides) {
    ArrayList<Double> llistaAmplitudsPartsCant = new ArrayList<>();
    int pasPartsCant, iteradorPartsCant, primeraPartCant, ultimaPartCant;
    double maximPartsCant1, maximPartsCant2, maximPartsCant3, maximPartsCant4, maximPartsCant5;
    double mitjanaPartsPotents;
    ArrayList<Boolean> negatius = new ArrayList<>();
    for(int i = 0; i < llistaCants.size(); i++) {
      //Apunta els cants amb un valor negatiu a l'inici (indicatiu d'anàlisi de Lyristes).
      if (llistaCants.get(i).getFirst() < 0) {
        llistaCants.set(i, new Pair(llistaCants.get(i).getFirst() * -1, llistaCants.get(i).getSecond()));
        negatius.add(true);
      }
      else
        negatius.add(false);

      //Si dura més de 250 ms
      if (llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst() > 2000) {
        maximPartsCant1 = 0;
        maximPartsCant2 = 0;
        maximPartsCant3 = 0;
        maximPartsCant4 = 0;
        maximPartsCant5 = 0;
        pasPartsCant = (int) Math.round((llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst()) / 20);

        //Fa una mitjana de les cinc vintenes parts més potents del cant
        for(int j = 0; j < 20; j++) {
```

```
            llistaAmplitudsPartsCant.add(trobaMitjanaAmbMaxims((int)
Math.round(4*(llistaCants.get(i).getFirst() + pasPartsCant * j)), 4*pasPartsCant,
amplitudsModificadesAbsolutesReduides, 4*pasPartsCant));
            if (llistaAmplitudsPartsCant.get(j) > maximPartsCant1) {
              maximPartsCant5 = maximPartsCant4;
              maximPartsCant4 = maximPartsCant3;
              maximPartsCant3 = maximPartsCant2;
              maximPartsCant2 = maximPartsCant1;
              maximPartsCant1 = llistaAmplitudsPartsCant.get(j);
            }
            else if (llistaAmplitudsPartsCant.get(j) > maximPartsCant2) {
              maximPartsCant5 = maximPartsCant4;
              maximPartsCant4 = maximPartsCant3;
              maximPartsCant3 = maximPartsCant2;
              maximPartsCant2 = llistaAmplitudsPartsCant.get(j);
            }
            else if (llistaAmplitudsPartsCant.get(j) > maximPartsCant3) {
              maximPartsCant5 = maximPartsCant4;
              maximPartsCant4 = maximPartsCant3;
              maximPartsCant3 = llistaAmplitudsPartsCant.get(j);
            }
            else if (llistaAmplitudsPartsCant.get(j) > maximPartsCant4) {
              maximPartsCant5 = maximPartsCant4;
              maximPartsCant4 = llistaAmplitudsPartsCant.get(j);
            }
            else if (llistaAmplitudsPartsCant.get(j) > maximPartsCant5) {
              maximPartsCant5 = llistaAmplitudsPartsCant.get(j);
            }
          }

          //Determina l'inici real del cant (considera que aquest no ha començat fins
que no arribi a 1/3.5 de la mitjana de les cinc vintenes parts més potents del cant).
          mitjanaPartsPotents = (maximPartsCant1 + maximPartsCant2 +
maximPartsCant3 + maximPartsCant4 + maximPartsCant5) / 5;
          iteradorPartsCant = 0;
          while (iteradorPartsCant < 20 &&
llistaAmplitudsPartsCant.get(iteradorPartsCant) < mitjanaPartsPotents / 3.5)
iteradorPartsCant++;
          primeraPartCant = iteradorPartsCant;
          ultimaPartCant = 19;

          //Si ha modificat l'inici del cant, l'actualitza a la llista
          if (primeraPartCant > 0) {
            llistaCants.set(i, new Pair(llistaCants.get(i).getFirst() + pasPartsCant *
primeraPartCant, llistaCants.get(i).getFirst() + pasPartsCant * (ultimaPartCant+1)));
          }
          llistaAmplitudsPartsCant.clear();
        }
      }
      for(int i = 0; i < negatius.size(); i++) {
        if (negatius.get(i)) {
          llistaCants.set(i, new Pair(llistaCants.get(i).getFirst() * -1,
llistaCants.get(i).getSecond()));
        }
      }
      return llistaCants;
    }

    /**
      @pre ---
      @post Retorna una llista de parelles que indiquen l'inici i el final dels trams on
considera que pot haver-hi un cant de cigala
      @return Una llista de parelles que indiquen l'inici i el final dels trams on considera
que pot haver-hi un cant de cigala
      @param amplitudsModificades Llista d'amplituds filtrades que permetran
determinar l'inici i el final dels cants de cigala dins de l'enregistrament
      @param amplitudsModificadesAbsolutes Llista d'amplituds filtrades i en valor
absolut que permetran determinar l'inici i el final dels cants de cigala dins de
l'enregistrament
      @param fs Freqüència de mostreig
      @param finestra Freqüència que utilitzarà per agrupar els valors
    */
    private static ArrayList<Pair<Double, Double> > trobaLlistaCants(ArrayList<Double>
amplitudsModificades, ArrayList<Double> amplitudsModificadesAbsolutes, int fs, int
finestra) {

      ArrayList<Integer> classificacioAmplituds = new ArrayList<>();

      //Les agrupa en grups de 5 ms.
      ArrayList<Double> amplitudsModificadesAbsolutesReduides =
redueixA(amplitudsModificadesAbsolutes,
(int) Math.round(amplitudsModificadesAbsolutes.size()/(2*fs*0.005)));

      double maximaAmplitud =
trobaMaximInferiorA(amplitudsModificadesAbsolutesReduides, trobaMitjana(0,
amplitudsModificadesAbsolutesReduides.size()-1,
amplitudsModificadesAbsolutesReduides)*7);
      double minimaAmplitud =
trobaMinim(amplitudsModificadesAbsolutesReduides);

      //Calcula quants trams de gravació hi ha per a cada nivell d'amplituds (hi ha 40
nivells diferents entre la mínima i la màxima)
      ArrayList<Double> particions = new ArrayList<>();

      double particio = 0;
      int nParticions = 40;

      for(int i = 1; i <= nParticions; i++) {
        particions.add(minimaAmplitud + (maximaAmplitud -
minimaAmplitud)*i/nParticions);
        classificacioAmplituds.add(0);
      }

      int iterador;
      for (Double amplitudsModificadesAbsolutesReduide :
amplitudsModificadesAbsolutesReduides) {
        iterador = 0;
        while (iterador < nParticions-1 && amplitudsModificadesAbsolutesReduide >
particions.get(iterador)) {
          iterador++;
        }
        classificacioAmplituds.set(iterador, classificacioAmplituds.get(iterador)+1);
      }

      //Determina l'inici i el final del que podem considerar cant a partir dels valors
d'amplitud.
      int nMin = Integer.MAX_VALUE;
      int nMax = 0;
      int nInici, nFinal;

      for(int i = 0; i < nParticions; i++) {
        if (i > 0 && classificacioAmplituds.get(i) > nMax) nMax =
classificacioAmplituds.get(i);
        if (classificacioAmplituds.get(i) < nMin) nMin = classificacioAmplituds.get(i);
      }

      int maxLocal;
      boolean trobat = false;
      int acumulat = 0;
      int totalValorsPotencialsCant = amplitudsModificadesAbsolutesReduides.size() -
classificacioAmplituds.get(0) - classificacioAmplituds.get(1) -
classificacioAmplituds.get(2);

      iterador = 3;
      while (iterador < nParticions - 1 && acumulat < totalValorsPotencialsCant/1.5 &&
!((nMax - classificacioAmplituds.get(iterador))/1.35 <
classificacioAmplituds.get(iterador) - nMin)) {
        acumulat += classificacioAmplituds.get(iterador);
        iterador++;
      }
      nInici = iterador;
      maxLocal = classificacioAmplituds.get(iterador);

      nMax = 0;
      for(int i = nInici; i < nParticions; i++) {
        if (classificacioAmplituds.get(i) > nMax) nMax = classificacioAmplituds.get(i);
      }

      if (maxLocal == nMax) trobat = true;
      iterador++;
      while (iterador < nParticions - 1 && !((nMax -
classificacioAmplituds.get(iterador))/1.5 > classificacioAmplituds.get(iterador) - nMin
&& maxLocal > 2*classificacioAmplituds.get(iterador) &&
classificacioAmplituds.get(iterador) < classificacioAmplituds.get(nInici) && trobat)) {
        if (classificacioAmplituds.get(iterador) > maxLocal) maxLocal =
classificacioAmplituds.get(iterador);
        if (maxLocal == nMax) trobat = true;
        iterador++;
      }
      nFinal = iterador-1;

      //Amplituds inferiors a la de nInici no formen part del cant.
      double llindar = ((maximaAmplitud - minimaAmplitud)*nInici/nParticions)*2.0;

      IdentificadorSilencis id = new IdentificadorSilencis(amplitudsModificades, fs);

      //Agrupa en finestres de 5 ms.
      ArrayList<Pair<Double, Double> > llistaSilencis = id.trobaSilencisEnTemps(llindar,
finestra);
      ArrayList<Pair<Double, Double> > cants = new ArrayList<>();

      //Treu "cants" massa curts.
      iterador = 1;
      while (iterador < llistaSilencis.size()) {
        if (llistaSilencis.get(iterador).getFirst() - llistaSilencis.get(iterador-1).getSecond()
< 8) {
          llistaSilencis.set(iterador-1, new Pair(llistaSilencis.get(iterador-1).getFirst(),
llistaSilencis.get(iterador).getSecond()));
          llistaSilencis.remove(iterador);
        }
        else iterador++;
```

```
    }

    //Treu silencis massa curts.
    iterador = 0;
    while (iterador < llistaSilencis.size()) {
      if (llistaSilencis.get(iterador).getSecond() - llistaSilencis.get(iterador).getFirst() < 8) {
        llistaSilencis.remove(iterador);
      }
      else iterador++;
    }

    if (llistaSilencis.size() > 0) {
      //Classifica els cants trobats segons la seva potència i elimina els que són massa fluixos, en comparació amb els altres, per poder considerar-se cants de la mateixa cigala.
      ArrayList<Integer> classificacioAmplitudsCants = new ArrayList<>();

      double maximaAmplitudCant;
      double minimaAmplitudCant;
      double maximLocal;
      double minimLocal;

      double limit = Double.MAX_VALUE;

      ArrayList<Double> particionsCants = new ArrayList<>();

      boolean fet = false;

      while ((llistaSilencis.size() > 2 || !fet) && (limit == Double.MAX_VALUE || (classificacioAmplitudsCants.get(nParticions-1) == 1 && classificacioAmplitudsCants.get(nParticions-2) == 0 && classificacioAmplitudsCants.get(nParticions-3) == 0))) {
        maximaAmplitudCant = Double.MIN_VALUE;
        minimaAmplitudCant = Double.MAX_VALUE;

        if (llistaSilencis.get(0).getFirst() > 0) {
          maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, 0, (int) Math.round(llistaSilencis.get(0).getFirst()*fs*2/1000));
          if (maximLocal > maximaAmplitudCant && maximLocal < limit) maximaAmplitudCant = maximLocal;
          minimLocal = trobaMinimLocal(amplitudsModificadesAbsolutes, 0, (int) Math.round(llistaSilencis.get(0).getFirst()*fs*2/1000));
          if (minimLocal < minimaAmplitudCant) minimaAmplitudCant = minimLocal;
        }
        for(int i = 0; i < llistaSilencis.size() - 1; i++) {
          maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, (int) Math.round(llistaSilencis.get(i).getSecond()*fs*2/1000), (int) Math.round(llistaSilencis.get(i+1).getFirst()*fs*2/1000));
          if (maximLocal > maximaAmplitudCant && maximLocal < limit) maximaAmplitudCant = maximLocal;
          minimLocal = trobaMinimLocal(amplitudsModificadesAbsolutes, (int) Math.round(llistaSilencis.get(i).getSecond()*fs*2/1000), (int) Math.round(llistaSilencis.get(i+1).getFirst()*fs*2/1000));
          if (minimLocal < minimaAmplitudCant) minimaAmplitudCant = minimLocal;
        }
        if ((amplitudsModificadesAbsolutes.size()-1)*1000/(fs*2) - llistaSilencis.get(llistaSilencis.size()-1).getSecond() > 10) {
          maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, (int) Math.round(llistaSilencis.get(llistaSilencis.size()-1).getSecond()*fs*2/1000), amplitudsModificadesAbsolutes.size()-1);
          if (maximLocal > maximaAmplitudCant && maximLocal < limit) maximaAmplitudCant = maximLocal;
          minimLocal = trobaMinimLocal(amplitudsModificadesAbsolutes, (int) Math.round(llistaSilencis.get(llistaSilencis.size()-1).getSecond()*fs*2/1000), amplitudsModificadesAbsolutes.size()-1);
          if (minimLocal < minimaAmplitudCant) minimaAmplitudCant = minimLocal;
        }

        limit = maximaAmplitudCant;


        particio = 0;
        nParticions = 40;

        particionsCants.clear();
        classificacioAmplitudsCants.clear();

        for(int i = 1; i <= nParticions; i++) {
          particionsCants.add(minimaAmplitudCant + (maximaAmplitudCant - minimaAmplitudCant)*i/nParticions);
          classificacioAmplitudsCants.add(0);
        }

        if (llistaSilencis.get(0).getFirst() > 0) {
          iterador = 0;
          maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, 0, (int) Math.round(llistaSilencis.get(0).getFirst()*fs*2/1000));
          while (iterador < nParticions - 1 && maximLocal > particionsCants.get(iterador)) {
            iterador++;
          }
          classificacioAmplitudsCants.set(iterador, classificacioAmplitudsCants.get(iterador)+1);
        }
        for(int i = 0; i < llistaSilencis.size() - 1; i++) {
          maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, (int) Math.round(llistaSilencis.get(i).getSecond()*fs*2/1000), (int) Math.round(llistaSilencis.get(i+1).getFirst()*fs*2/1000));
          iterador = 0;
          while (iterador < nParticions - 1 && maximLocal > particionsCants.get(iterador)) {
            iterador++;
          }
          classificacioAmplitudsCants.set(iterador, classificacioAmplitudsCants.get(iterador)+1);
        }

        if ((amplitudsModificadesAbsolutes.size()-1)*1000/(fs*2) - llistaSilencis.get(llistaSilencis.size()-1).getSecond() > 10) {
          maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, (int) Math.round(llistaSilencis.get(llistaSilencis.size()-1).getSecond()*fs*2/1000), amplitudsModificadesAbsolutes.size()-1);
          iterador = 0;
          while (iterador < nParticions - 1 && maximLocal > particionsCants.get(iterador)) {
            iterador++;
          }
          classificacioAmplitudsCants.set(iterador, classificacioAmplitudsCants.get(iterador)+1);
        }

        fet = true;
      }

      //Troba un llindar mínim a partir del qual considerarà que els cants són vàlids.
      nMin = Integer.MAX_VALUE;
      nMax = 0;

      for(int i = 0; i < nParticions; i++) {
        if (i > 0 && classificacioAmplitudsCants.get(i) > nMax) nMax = classificacioAmplitudsCants.get(i);
        if (classificacioAmplitudsCants.get(i) < nMin) nMin = classificacioAmplitudsCants.get(i);
      }

      iterador = 1;
      while (iterador < nParticions - 1 && classificacioAmplitudsCants.get(iterador) <= classificacioAmplitudsCants.get(iterador-1)) {
        iterador++;
      }
      nInici = iterador;



      if (llistaSilencis.get(0).getFirst() > 0) {
        iterador = 0;
        maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, 0, (int) Math.round(llistaSilencis.get(0).getFirst()*fs*2/1000));
        while (iterador < nParticions - 1 && maximLocal > particionsCants.get(iterador)) {
          iterador++;
        }
        if (iterador < nInici) {
          llistaSilencis.set(0, new Pair(0.0, llistaSilencis.get(0).getSecond()));
        }
      }
      for(int i = 0; i < llistaSilencis.size() - 1; i++) {
        maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, (int) Math.round(llistaSilencis.get(i).getSecond()*fs*2/1000), (int) Math.round(llistaSilencis.get(i+1).getFirst()*fs*2/1000));
        iterador = 0;
        while (iterador < nParticions - 1 && maximLocal > particionsCants.get(iterador)) {
          iterador++;
        }
        if (iterador < nInici) {
          llistaSilencis.set(i, new Pair(llistaSilencis.get(i).getFirst(), llistaSilencis.get(i+1).getSecond()));
          llistaSilencis.remove(i+1);
          i--;
        }
      }

      if ((amplitudsModificadesAbsolutes.size()-1)*1000/(fs*2) - llistaSilencis.get(llistaSilencis.size()-1).getSecond() > 10) {
        maximLocal = trobaMaximLocal(amplitudsModificadesAbsolutes, (int) Math.round(llistaSilencis.get(llistaSilencis.size()-1).getSecond()*fs*2/1000), amplitudsModificadesAbsolutes.size()-1);
        iterador = 0;
        while (iterador < nParticions - 1 && maximLocal > particionsCants.get(iterador)) {
          iterador++;
```

```
            }
            if (iterador < nInici) {
              llistaSilencis.set(llistaSilencis.size()-1, new
Pair(llistaSilencis.get(llistaSilencis.size()-1).getFirst(),
(amplitudsModificadesAbsolutes.size()-1)*1000/(2*fs)));
            }
          }

          //Retorna la llista de cants a partir de la llista de silencis que tenia fins ara.
          if (llistaSilencis.get(0).getFirst() > 0) {
            cants.add(new Pair(0.0, llistaSilencis.get(0).getFirst()));
          }
          for(int i = 0; i < llistaSilencis.size() - 1; i++) {
            cants.add(new Pair(llistaSilencis.get(i).getSecond(),
llistaSilencis.get(i+1).getFirst()));
          }

          if ((amplitudsModificadesAbsolutes.size()-1)*1000.0/(fs*2) -
llistaSilencis.get(llistaSilencis.size()-1).getSecond() > 10) {
            cants.add(new Pair(llistaSilencis.get(llistaSilencis.size()-1).getSecond(),
(amplitudsModificadesAbsolutes.size()-1)*1000.0/(fs*2)));
          }

        }
        else {
          cants.add(new Pair(0.0, (amplitudsModificadesAbsolutes.size()-
1)*1000.0/(fs*2)));
        }
        return cants;
      }

      /**
        @pre ---
        @post Retorna una llista de parelles que indiquen l'inici i el final dels cants
concrets de la cigala enregistrada
        @return Una llista de parelles que indiquen l'inici i el final dels cants concrets de
la cigala enregistrada
        @param amplitudsModificadesAbsolutesReduides Llista d'amplituds filtrades, en
valor absolut i escalades que permetran determinar l'inici i el final dels cants de cigala
dins de l'enregistrament
        @param fs Freqüència de mostreig
        @param inici Moment del fitxer a partir del qual hem de començar a analitzar
        @param fi Moment del fitxer fins al qual hem d'analitzar
        @param duradaMinimaSilenci Durada mínima que considero que pot tenir un
silenci entre dos cants
        @param coef Coeficient que m'indica la diferència proporcional mínima entre dos
trams contigus del fitxer per considerar que el primer no és cant i el segon sí en fer una
anàlisi del moment d'inici aproximat del cant
        @param coef2 Coeficient que m'indica la diferència proporcional mínima entre
dos trams contigus del fitxer per considerar que el primer és cant i el segon no en fer
una anàlisi genèrica del moment final aproximat del cant
        @param coef3 Coeficient que m'indica la diferència proporcional mínima entre
dos trams contigus del fitxer per considerar que el primer no és cant i el segon sí en fer
una anàlisi precisa del moment d'inici del cant. En cas que sigui negatiu, m'indica que
he de fer una anàlisi orientada a determinar l'inici i el final dels cants d'una Lyristes
plebejus.
      */
      private static ArrayList<Pair<Double, Double> >
trobaCantsConcrets(ArrayList<Double> amplitudsModificadesAbsolutesReduides, int
fs, double inici, double fi, int duradaMinimaSilenci, double coef, double coef2, double
coef3) {
        ArrayList<Pair<Double, Double> > llistaCants = new ArrayList< >();
        ArrayList<Pair<Double, Double> > llistaCantsPendents = new ArrayList< >();

        boolean analisiLyristes = false;
        ArrayList<Integer> inicisCantContinu = new ArrayList<>();
        ArrayList<Integer> finalsCantContinu = new ArrayList<>();

        if (coef3 < 0) {
          coef3 = -coef3;
          analisiLyristes = true;
        }

        int posicioInici = (int) Math.round(inici*4.0);
        int posicioFi = (int) Math.round(fi*4.0); //Indica la posició de la llista
AmplitudsModificades on s'acaba el cant.

        int iniciAux = posicioInici + duradaMinimaSilenci*4; //Avança
4*duradaMinimaSilenci milisegons respecte a l'inici del cant per poder fer
comparacions.
        int finalAux = iniciAux;

        int durada, durada2;

        //Durades en milisegons dels trams que agrupo per determinar els inicis i finals
dels cants
        durada = (int) Math.round(Math.min(duradaMinimaSilenci*4.0, posicioFi-
finalAux-1));
        durada2 = Math.min(durada, 5);

       //En lloc de recalcular les mitjanes cada cop, les actualitza, per fer-ho més eficient.
        Pair<Double, Double> mitjanaIMaximCantActual =
actualitzaMitjanaAmbMaxims(iniciAux, finalAux-iniciAux,
amplitudsModificadesAbsolutesReduides, finalAux-iniciAux, 0.0, 0.0, 1, false, false);
        Pair<Double, Double> mitjanaIMaximPostCantActual =
actualitzaMitjanaAmbMaxims(finalAux, durada,
amplitudsModificadesAbsolutesReduides, 40, 0.0, 0.0, 1, false, false);
        Pair<Double, Double> mitjanaIMaximCantActual2 =
actualitzaMitjanaAmbMaxims(posicioInici, finalAux-posicioInici,
amplitudsModificadesAbsolutesReduides, finalAux-posicioInici, 0.0, 0.0, 1, false, false);
        Pair<Double, Double> mitjanaIMaximPostCantActual2 =
actualitzaMitjanaAmbMaxims(finalAux, durada2,
amplitudsModificadesAbsolutesReduides, durada2, 0.0, 0.0, 1, false, false);
        boolean continua = !((mitjanaIMaximCantActual.getFirst() == 0.0 &&
mitjanaIMaximCantActual.getSecond() == 0.0 && finalAux != iniciAux) ||
            (mitjanaIMaximPostCantActual.getFirst() == 0.0 &&
mitjanaIMaximPostCantActual.getSecond() == 0.0) ||
            (mitjanaIMaximCantActual2.getFirst() == 0.0 &&
mitjanaIMaximCantActual2.getSecond() == 0.0) ||
            (mitjanaIMaximPostCantActual2.getFirst() == 0.0 &&
mitjanaIMaximPostCantActual2.getSecond() == 0.0) ||
            (mitjanaIMaximCantActual.getFirst() >
mitjanaIMaximPostCantActual.getFirst()*coef &&
mitjanaIMaximCantActual2.getFirst() >
mitjanaIMaximPostCantActual2.getFirst()*coef2)
            );

        //Millor no calcular la mitjana cada cop
        //Va avançant fins que troba el final del primer cant.
        while (finalAux < posicioFi - durada && continua) {
          finalAux++;
          durada = (int) Math.round(Math.min(duradaMinimaSilenci*4.0, posicioFi-
finalAux-1));
          durada2 = Math.min(durada, 5);

          if (finalAux < posicioFi - durada) {
            mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, finalAux-
iniciAux, amplitudsModificadesAbsolutesReduides, finalAux-iniciAux,
mitjanaIMaximCantActual.getFirst(), mitjanaIMaximCantActual.getSecond(), 1, true,
false);
            mitjanaIMaximPostCantActual = actualitzaMitjanaAmbMaxims(finalAux,
durada, amplitudsModificadesAbsolutesReduides, 40,
mitjanaIMaximPostCantActual.getFirst(), mitjanaIMaximPostCantActual.getSecond(),
1, false, false);
            mitjanaIMaximCantActual2 = actualitzaMitjanaAmbMaxims(posicioInici,
finalAux-posicioInici, amplitudsModificadesAbsolutesReduides, finalAux-posicioInici,
mitjanaIMaximCantActual2.getFirst(), mitjanaIMaximCantActual2.getSecond(), 1, true,
false);
            mitjanaIMaximPostCantActual2 = actualitzaMitjanaAmbMaxims(finalAux,
durada2, amplitudsModificadesAbsolutesReduides, durada2,
mitjanaIMaximPostCantActual2.getFirst(),
mitjanaIMaximPostCantActual2.getSecond(), 1, false, false);
            if ((mitjanaIMaximCantActual.getFirst() == 0.0 &&
mitjanaIMaximCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximPostCantActual.getFirst() == 0.0 &&
mitjanaIMaximPostCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximCantActual2.getFirst() == 0.0 &&
mitjanaIMaximCantActual2.getSecond() == 0.0) ||
              (mitjanaIMaximPostCantActual2.getFirst() == 0.0 &&
mitjanaIMaximPostCantActual2.getSecond() == 0.0) ||
              (mitjanaIMaximCantActual.getFirst() >
mitjanaIMaximPostCantActual.getFirst()*coef &&
mitjanaIMaximCantActual2.getFirst() >
mitjanaIMaximPostCantActual2.getFirst()*coef2)
              ) continua = false;
          }
        }

        int finalPrimerCant = finalAux;
        int iniciAbansAjustament;

        boolean primerCantAfegit = false;

        double iniciOriginal;


        while (finalAux < posicioFi) {
          //Determina l'inici del cant.
          mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, 5,
amplitudsModificadesAbsolutesReduides, 5, 0.0, 0.0, 1, false, false);
          Pair<Double, Double> mitjanaIMaximPreCantActual =
actualitzaMitjanaAmbMaxims(iniciAux - duradaMinimaSilenci*4,
(duradaMinimaSilenci-1)*4, amplitudsModificadesAbsolutesReduides, -40, 0.0, 0.0, 1,
false, false);

          continua = !((mitjanaIMaximCantActual.getFirst() == 0.0 &&
mitjanaIMaximCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximPreCantActual.getFirst() == 0.0 &&
mitjanaIMaximPreCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximCantActual.getFirst() >
mitjanaIMaximPreCantActual.getFirst()*coef)
              );
```

```
        while (iniciAux < posicioFi && continua) {
          iniciAux++;
          if (iniciAux < posicioFi) {
            mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, 5, amplitudsModificadesAbsolutesReduides, 5, mitjanaIMaximCantActual.getFirst(), mitjanaIMaximCantActual.getSecond(), 1, false, false);
            mitjanaIMaximPreCantActual = actualitzaMitjanaAmbMaxims(iniciAux - duradaMinimaSilenci*4, (duradaMinimaSilenci-1)*4, amplitudsModificadesAbsolutesReduides, -40, mitjanaIMaximPreCantActual.getFirst(), mitjanaIMaximPreCantActual.getSecond(), 1, false, false);
            if ((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximPreCantActual.getFirst() == 0.0 && mitjanaIMaximPreCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximPreCantActual.getFirst()*coef)
              ) continua = false;
          }
        }

        if (iniciAux < posicioFi) {
          if (!primerCantAfegit) {
            if (finalPrimerCant < iniciAux && finalPrimerCant - posicioInici > 4.0*10) {
              llistaCants.add(new Pair(posicioInici/4.0, finalPrimerCant/4.0));
            }
            //Li dóna una segona oportunitat (pot ser que hagi trobat l'inici del segon cant però no el final del primer, cas en què el final del primer cauria més tard que el principi del segon i, per tant, no l'afegiria.
            else if (finalPrimerCant > iniciAux) {
              //Determina el final del primer cant.
              int copiaFinalPrimerCant = finalPrimerCant;
              finalPrimerCant = iniciAux - 1;
              mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(finalPrimerCant - 5, 5, amplitudsModificadesAbsolutesReduides, 5, 0.0, 0.0, 1, false, false);
              mitjanaIMaximPostCantActual = actualitzaMitjanaAmbMaxims(finalPrimerCant, (duradaMinimaSilenci-1)*4, amplitudsModificadesAbsolutesReduides, 40, 0.0, 0.0, 1, false, false);

              continua = !((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
                  (mitjanaIMaximPostCantActual.getFirst() == 0.0 && mitjanaIMaximPostCantActual.getSecond() == 0.0) ||
                  (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximPostCantActual.getFirst()*coef*0.75)
                  );
              while (finalPrimerCant > posicioInici + 5 && continua) {
                finalPrimerCant--;
                if (finalPrimerCant > posicioInici + 5) {
                  mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(finalPrimerCant - 5, 5, amplitudsModificadesAbsolutesReduides, 5, mitjanaIMaximCantActual.getFirst(), mitjanaIMaximCantActual.getSecond(), -1, false, false);
                  mitjanaIMaximPostCantActual = actualitzaMitjanaAmbMaxims(finalPrimerCant, (duradaMinimaSilenci-1)*4, amplitudsModificadesAbsolutesReduides, 40, mitjanaIMaximPostCantActual.getFirst(), mitjanaIMaximPostCantActual.getSecond(), -1, false, false);
                  if ((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
                    (mitjanaIMaximPostCantActual.getFirst() == 0.0 && mitjanaIMaximPostCantActual.getSecond() == 0.0) ||
                    (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximPostCantActual.getFirst()*coef*0.75)
                    ) continua = false;
                }
              }
              //Si no ha trobat cap final abans és que simplement el final que havia trobat en un principi´ja era el final del primer cant.
              if (finalPrimerCant < copiaFinalPrimerCant && finalPrimerCant > posicioInici && finalPrimerCant < iniciAux && finalPrimerCant - posicioInici > 4.0*10) {
                llistaCants.add(new Pair(posicioInici/4.0, finalPrimerCant/4.0));
              }
            }
            primerCantAfegit = true;
          }
          finalAux = iniciAux+1;

          durada = (int) Math.round(Math.min(duradaMinimaSilenci*4.0, posicioFi-finalAux-1));
          durada2 = Math.min(durada, 5);

          //Determina el final del cant.
          mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, finalAux-iniciAux, amplitudsModificadesAbsolutesReduides, finalAux-iniciAux, 0.0, 0.0, 1, true, false);
          mitjanaIMaximPostCantActual = actualitzaMitjanaAmbMaxims(finalAux, durada, amplitudsModificadesAbsolutesReduides, Math.min(durada, 40), 0.0, 0.0, 1, false, false);
          mitjanaIMaximPostCantActual2 = actualitzaMitjanaAmbMaxims(finalAux, durada2, amplitudsModificadesAbsolutesReduides, durada2, 0.0, 0.0, 1, false, false);

          continua = !(((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximPostCantActual.getFirst() == 0.0 && mitjanaIMaximPostCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximPostCantActual2.getFirst() == 0.0 && mitjanaIMaximPostCantActual2.getSecond() == 0.0) ||
              (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximPostCantActual.getFirst()*coef && mitjanaIMaximCantActual.getFirst() > mitjanaIMaximPostCantActual2.getFirst()*coef2)
              ));
          while (finalAux < posicioFi - durada && continua) {
            finalAux++;
            durada = (int) Math.round(Math.min(duradaMinimaSilenci*4.0, posicioFi-finalAux-1));
            durada2 = Math.min(durada, 5);
            if (finalAux < posicioFi - durada) {
              //Té en compte si les durades han disminuït.
              mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, finalAux-iniciAux, amplitudsModificadesAbsolutesReduides, finalAux-iniciAux, mitjanaIMaximCantActual.getFirst(), mitjanaIMaximCantActual.getSecond(), 1, true, false);
              mitjanaIMaximPostCantActual = actualitzaMitjanaAmbMaxims(finalAux, durada, amplitudsModificadesAbsolutesReduides, Math.min(durada, 40), mitjanaIMaximPostCantActual.getFirst(), mitjanaIMaximPostCantActual.getSecond(), 1, false, durada < duradaMinimaSilenci*4.0);
              mitjanaIMaximPostCantActual2 = actualitzaMitjanaAmbMaxims(finalAux, durada2, amplitudsModificadesAbsolutesReduides, durada2, mitjanaIMaximPostCantActual2.getFirst(), mitjanaIMaximPostCantActual2.getSecond(), 1, false, durada2 < 5);
              if ((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
                (mitjanaIMaximPostCantActual.getFirst() == 0.0 && mitjanaIMaximPostCantActual.getSecond() == 0.0) ||
                (mitjanaIMaximPostCantActual2.getFirst() == 0.0 && mitjanaIMaximPostCantActual2.getSecond() == 0.0) ||
                (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximPostCantActual.getFirst()*coef && mitjanaIMaximCantActual.getFirst() > mitjanaIMaximPostCantActual2.getFirst()*coef2)
                ) continua = false;
            }
          }

          //Si el cant trobat dura més de 12 milisegons
          if (finalAux - iniciAux > 4.0*12) {
            iniciOriginal = iniciAux;

            //Ajusta l'inici del cant, perquè em marqui ben bé quan aquest comença.
            iniciAbansAjustament = iniciAux;
            durada = (int) Math.round(Math.min(10*4.0, finalAux-iniciAux));

            mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, durada, amplitudsModificadesAbsolutesReduides, durada, 0.0, 0.0, 1, false, false);
            mitjanaIMaximCantActual2 = actualitzaMitjanaAmbMaxims(iniciAbansAjustament-duradaMinimaSilenci*4, duradaMinimaSilenci*4, amplitudsModificadesAbsolutesReduides, -40, 0.0, 0.0, 1, false, false);
            continua = !(((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
              (mitjanaIMaximCantActual2.getFirst() == 0.0 && mitjanaIMaximCantActual2.getSecond() == 0.0) ||
              (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximCantActual2.getFirst()*coef2)
              ));
            while (iniciAux < finalAux - durada && continua) {
              iniciAux++;
              durada = (int) Math.round(Math.min(10*4.0, finalAux-iniciAux));
              if (iniciAux < finalAux - durada) {
                mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, durada, amplitudsModificadesAbsolutesReduides, durada, mitjanaIMaximCantActual.getFirst(), mitjanaIMaximCantActual.getSecond(), 1, false, durada < 10*4.0);
                mitjanaIMaximCantActual2 = actualitzaMitjanaAmbMaxims(iniciAbansAjustament-duradaMinimaSilenci*4, duradaMinimaSilenci*4, amplitudsModificadesAbsolutesReduides, -40, mitjanaIMaximCantActual2.getFirst(), mitjanaIMaximCantActual2.getSecond(), 1, false, false);
                if (((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
                  (mitjanaIMaximCantActual2.getFirst() == 0.0 && mitjanaIMaximCantActual2.getSecond() == 0.0) ||
                  (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximCantActual2.getFirst()*coef2)
                  )) continua = false;
              }
            }

            iniciAbansAjustament = iniciAux;
```

```
            if (analisiLyristes) inicisCantContinu.add(iniciAux);

            mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, durada, amplitudsModificadesAbsolutesReduides, durada, 0.0, 0.0, 1, false, false);
            mitjanaIMaximCantActual2 = actualitzaMitjanaAmbMaxims(iniciAbansAjustament, iniciAux - iniciAbansAjustament, amplitudsModificadesAbsolutesReduides, iniciAux-iniciAbansAjustament, 0.0, 0.0, 1, false, false);
            Pair<Double, Double> mitjanaIMaximCantActual3 = actualitzaMitjanaAmbMaxims(iniciAux, finalAux-iniciAux, amplitudsModificadesAbsolutesReduides, finalAux-iniciAux, 0.0, 0.0, 1, false, false);
            boolean ajustaInici = (((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
                    (mitjanaIMaximCantActual2.getFirst() == 0.0 && mitjanaIMaximCantActual2.getSecond() == 0.0) ||
                    (mitjanaIMaximCantActual3.getFirst() == 0.0 && mitjanaIMaximCantActual3.getSecond() == 0.0) ||
                    (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximCantActual2.getFirst()*coef2 && mitjanaIMaximCantActual3.getFirst() > mitjanaIMaximCantActual2.getFirst()*coef3)
                    ));
            boolean ajustat = false;

            int nombreMs = Math.min(finalAux - iniciAux, 100*4);
            int nombreMs2 = Math.min(iniciAux - posicioInici, 100*4);
            double diferenciaIniciCant = actualitzaMitjanaAmbMaxims(iniciAux, nombreMs, amplitudsModificadesAbsolutesReduides, nombreMs, 0.0, 0.0, 1, false, false).getFirst() / actualitzaMitjanaAmbMaxims(iniciAux - nombreMs2, nombreMs2 - 4, amplitudsModificadesAbsolutesReduides, -nombreMs2, 0.0, 0.0, 1, false, false).getFirst();
            nombreMs2 = Math.min(posicioFi - finalAux, 100*4);
            double diferenciaFinalCant = actualitzaMitjanaAmbMaxims(finalAux - nombreMs, nombreMs, amplitudsModificadesAbsolutesReduides, nombreMs, 0.0, 0.0, 1, true, false).getFirst() / actualitzaMitjanaAmbMaxims(finalAux, nombreMs2, amplitudsModificadesAbsolutesReduides, nombreMs2, 0.0, 0.0, 1, false, false).getFirst();
            //Tibicina garricola: diferenciaIniciCant ~2.5, diferenciaFinalCant ~5
            while (iniciAux < finalAux - durada && diferenciaIniciCant + diferenciaFinalCant < 6) {
              //Si hi ha una pujada brusca i a partir d'allà les amplituds són molt majors que fins a aquell moment, ajusta l'inici del cant a aquest punt
              if (ajustaInici) {
                iniciAbansAjustament = iniciAux;
                ajustat = true;
              }

              iniciAux++;
              durada = (int) Math.round(Math.min(10*4.0, finalAux-iniciAux));

              //Si el cant arriba a 10 milisegons, determina si s'ha d'ajustar l'inici o no.
              if (iniciAux < finalAux - durada) {
                if (ajustat) {
                  ajustat = false;
                  mitjanaIMaximCantActual2 = actualitzaMitjanaAmbMaxims(iniciAbansAjustament, iniciAux - iniciAbansAjustament, amplitudsModificadesAbsolutesReduides, iniciAux-iniciAbansAjustament, 0.0, 0.0, 1, false, false);
                }
                else {
                  mitjanaIMaximCantActual2 = actualitzaMitjanaAmbMaxims(iniciAbansAjustament, iniciAux - iniciAbansAjustament, amplitudsModificadesAbsolutesReduides, iniciAux-iniciAbansAjustament, mitjanaIMaximCantActual2.getFirst(), mitjanaIMaximCantActual2.getSecond(), 1, true, false);
                }
                mitjanaIMaximCantActual = actualitzaMitjanaAmbMaxims(iniciAux, durada, amplitudsModificadesAbsolutesReduides, durada, mitjanaIMaximCantActual.getFirst(), mitjanaIMaximCantActual.getSecond(), 1, false, false);
                mitjanaIMaximCantActual3 = actualitzaMitjanaAmbMaxims(iniciAux, finalAux-iniciAux, amplitudsModificadesAbsolutesReduides, finalAux-iniciAux, mitjanaIMaximCantActual3.getFirst(), mitjanaIMaximCantActual3.getSecond(), 1, false, true);


                ajustaInici = (iniciAux > iniciAbansAjustament + 10*4.0 &&
                  ((mitjanaIMaximCantActual.getFirst() == 0.0 && mitjanaIMaximCantActual.getSecond() == 0.0) ||
                  (mitjanaIMaximCantActual2.getFirst() == 0.0 && mitjanaIMaximCantActual2.getSecond() == 0.0) ||
                  (mitjanaIMaximCantActual3.getFirst() == 0.0 && mitjanaIMaximCantActual3.getSecond() == 0.0) ||
                  (mitjanaIMaximCantActual.getFirst() > mitjanaIMaximCantActual2.getFirst()*coef2 && mitjanaIMaximCantActual3.getFirst() > mitjanaIMaximCantActual2.getFirst()*coef3)
                  ));
              }
            }

            iniciAux = iniciAbansAjustament;

            if (analisiLyristes) finalsCantContinu.add(iniciAux);

            if (finalAux - iniciAux > 4.0*12) {
              llistaCants.add(new Pair(iniciAux/4.0, finalAux/4.0));
            }

            //Si està fent una anàlisi per a Lyristes plebejus, ja l'haurà afegit a inicisCantContinu i finalsCantContinu.
            if (iniciAux - iniciOriginal > 100 && !analisiLyristes) {
              llistaCantsPendents.add(new Pair(iniciOriginal/4.0, iniciAux/4.0));
            }
          }
        }
        else { //El final del cant correspon al final del tram del fitxer que havia d'analitzar.
          if (llistaCants.isEmpty()) llistaCants.add(new Pair(inici, fi));
          finalAux = posicioFi;
        }

        iniciAux = finalAux + 1;
      }

      double mitjanaAmplitudCant = 0;

      for (Pair<Double, Double> llistaCant : llistaCants) {
        mitjanaAmplitudCant += trobaMitjanaAmbMaxims((int) Math.round(4 * llistaCant.getFirst()), (int) Math.round(4 * (llistaCant.getSecond() - llistaCant.getFirst())), amplitudsModificadesAbsolutesReduides, (int) Math.round(4 * (llistaCant.getSecond() - llistaCant.getFirst())));
      }

      mitjanaAmplitudCant /= llistaCants.size();

      int indexInsercio;

      //Si té prou amplitud, l'afegeix a la llista de cants.
      for (Pair<Double, Double> llistaCantsPendent : llistaCantsPendents) {
        if (trobaMitjanaAmbMaxims((int) Math.round(4 * llistaCantsPendent.getFirst()), (int) Math.round(4 * (llistaCantsPendent.getSecond() - llistaCantsPendent.getFirst())), amplitudsModificadesAbsolutesReduides, (int) Math.round(4 * (llistaCantsPendent.getSecond() - llistaCantsPendent.getFirst()))) > mitjanaAmplitudCant*0.4) {
          indexInsercio = trobaPosicioInsercioOrdenada(llistaCants, llistaCantsPendent);
          llistaCants.add(indexInsercio, llistaCantsPendent);
        }
      }

      double desviacioEstandard = trobaDesviacioEstandard(llistaCants);

      if (desviacioEstandard > 0.5) {

        int iniciGrup, fiGrup;
        ArrayList<ArrayList<Pair<Double, Double> > > grupsCants = new ArrayList< >();
        ArrayList<Pair<Double, Double> > aux;
        boolean afegirUltim = true;

        //Separa els cants en grups.
        for(int i = 1; i < llistaCants.size(); i++) {
          //Agrupa els cants si són contigus i semblants entre ells.
          iniciGrup = i-1;
          while (i < llistaCants.size() && llistaCants.get(i-1).getSecond() - llistaCants.get(i-1).getFirst() < (llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst())*2 && llistaCants.get(i-1).getSecond() - llistaCants.get(i-1).getFirst() > (llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst())/2) {
            i++;
          }
          fiGrup = i-1;

          if (i == llistaCants.size() && llistaCants.get(i-2).getSecond() - llistaCants.get(i-2).getFirst() < (llistaCants.get(i-1).getSecond() - llistaCants.get(i-1).getFirst())*2 && llistaCants.get(i-2).getSecond() - llistaCants.get(i-2).getFirst() > (llistaCants.get(i-1).getSecond() - llistaCants.get(i-1).getFirst())/2) {
            afegirUltim = false;
          }

          aux = new ArrayList<>();
          for(int j = iniciGrup; j <= fiGrup; j++) {
            aux.add(llistaCants.get(j));
          }

          grupsCants.add(aux);
        }

        if (afegirUltim) {
          aux = new ArrayList<>();
          aux.add(llistaCants.get(llistaCants.size()-1));
          grupsCants.add(aux);
        }

        ArrayList<Pair<Double, Double> > llistaDefinitiva = new ArrayList<>();
```

```
        ArrayList<Pair<Double, Double> > llistaPerAgrupar = new ArrayList<>();

        //Decideix si els grups de cants són correctes o bé no ho són i s'han d'agrupar.
        boolean correcte;
        for (ArrayList<Pair<Double, Double>> grupsCant : grupsCants) {
          correcte = true;
          if (grupsCant.size() < 3 || trobaDesviacioEstandard(grupsCant) > 0.35) {
            correcte = false;
          }
          //Mira que l'últim cant no sigui massa llarg perquè al bucle posterior no ho mirarà.
          if (grupsCant.get(grupsCant.size() - 1).getSecond() - grupsCant.get(grupsCant.size() - 1).getFirst() > 2000) {
            correcte = false;
          }
          int it = 0;
          while (correcte && it < grupsCant.size() - 1) {
            if (grupsCant.get(it+1).getFirst() - grupsCant.get(it).getSecond() > 400 || grupsCant.get(it).getSecond() - grupsCant.get(it).getFirst() > 2000) {
              correcte = false;
            }
            it++;
          }
          if (correcte) {
            for(Pair<Double, Double> grupsCant1: grupsCant) {
              llistaDefinitiva.add(grupsCant1);
            }
          }
          else {
            for(Pair<Double, Double> grupsCant1: grupsCant) {
              llistaPerAgrupar.add(grupsCant1);
            }
          }
        }

        ArrayList<Pair<Double, Double> > llistaAgrupats;
        if (!analisiLyristes) llistaAgrupats = agrupaCants(llistaPerAgrupar);
        else {
          llistaAgrupats = new ArrayList<>();
          llistaAgrupats.addAll(llistaPerAgrupar);
        }

        for (Pair<Double, Double> llistaAgrupat : llistaAgrupats) {
          indexInsercio = trobaPosicioInsercioOrdenada(llistaDefinitiva, llistaAgrupat);
          llistaDefinitiva.add(indexInsercio, llistaAgrupat);
        }

        //Elimina els cants massa baixos.
        double mitjanaAmplitudCants = 0.0;
        double tempsTotalCant = 0.0;

        for (Pair<Double, Double> llistaDefinitiva1 : llistaDefinitiva) {
          mitjanaAmplitudCants += (llistaDefinitiva1.getSecond() - llistaDefinitiva1.getFirst()) * 4 * trobaMitjanaAmbMaxims((int) Math.round(llistaDefinitiva1.getFirst() * 4), (int) Math.round((llistaDefinitiva1.getSecond() - llistaDefinitiva1.getFirst()) * 4), amplitudsModificadesAbsolutesReduides, (int) Math.round((llistaDefinitiva1.getSecond() - llistaDefinitiva1.getFirst()) * 4));
          tempsTotalCant += (llistaDefinitiva1.getSecond() - llistaDefinitiva1.getFirst()) * 4;
        }

        mitjanaAmplitudCants /= tempsTotalCant;


        for(int i = 0; i < llistaDefinitiva.size(); i++) {
          if (trobaMitjanaAmbMaxims((int) Math.round(llistaDefinitiva.get(i).getFirst()*4), (int) Math.round((llistaDefinitiva.get(i).getSecond() - llistaDefinitiva.get(i).getFirst())*4), amplitudsModificadesAbsolutesReduides, (int) Math.round((llistaDefinitiva.get(i).getSecond() - llistaDefinitiva.get(i).getFirst())*4)) < mitjanaAmplitudCants*0.07) {
            llistaDefinitiva.remove(i);
          }
        }

        //No vull que l'agrupi amb el següent, perquè aquest és el cant continu i va millor tenir-lo a part per mirar-ne el NGP.
        if (analisiLyristes) {
          for(int i = 0; i < inicisCantContinu.size(); i++) {
            if (finalsCantContinu.get(i) - inicisCantContinu.get(i) > 1500*4 && trobaMitjanaAmbMaxims(inicisCantContinu.get(i), finalsCantContinu.get(i)-inicisCantContinu.get(i), amplitudsModificadesAbsolutesReduides, finalsCantContinu.get(i)-inicisCantContinu.get(i)) > mitjanaAmplitudCants*0.125) {
              indexInsercio = trobaPosicioInsercioOrdenada(llistaDefinitiva, new Pair(inicisCantContinu.get(i)/4.0, finalsCantContinu.get(i)/4.0));
              llistaDefinitiva.add(indexInsercio, new Pair(inicisCantContinu.get(i)/4.0, finalsCantContinu.get(i)/4.0));
            }
          }
        }

        //Ajusta els cants al seu inici i final real d'una manera més precisa.
        return ajustaCants(llistaDefinitiva, amplitudsModificadesAbsolutesReduides);
      }

      //Elimina els cants massa baixos (encara que ja s'hagi fet si té una desviació estàndard elevada, tant els cants com la mitjana d'amplitud poden haver canviat).
      double mitjanaAmplitudCants = 0.0;
      double tempsTotalCant = 0.0;

      for (Pair<Double, Double> llistaCant : llistaCants) {
        mitjanaAmplitudCants += (llistaCant.getSecond() - llistaCant.getFirst()) * 4 * trobaMitjanaAmbMaxims((int) Math.round(llistaCant.getFirst() * 4), (int) Math.round((llistaCant.getSecond() - llistaCant.getFirst()) * 4), amplitudsModificadesAbsolutesReduides, (int) Math.round((llistaCant.getSecond() - llistaCant.getFirst()) * 4));
        tempsTotalCant += (llistaCant.getSecond() - llistaCant.getFirst()) * 4;
      }

      mitjanaAmplitudCants /= tempsTotalCant;


      for(int i = 0; i < llistaCants.size(); i++) {
        if (trobaMitjanaAmbMaxims((int) Math.round(llistaCants.get(i).getFirst()*4), (int) Math.round((llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst())*4), amplitudsModificadesAbsolutesReduides, (int) Math.round((llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst())*4)) < mitjanaAmplitudCants*0.07) {
          llistaCants.remove(i);
        }
      }

      if (analisiLyristes) {
        for(int i = 0; i < inicisCantContinu.size(); i++) {
          if (finalsCantContinu.get(i) - inicisCantContinu.get(i) > 1500*4 && trobaMitjanaAmbMaxims(inicisCantContinu.get(i), finalsCantContinu.get(i)-inicisCantContinu.get(i), amplitudsModificadesAbsolutesReduides, finalsCantContinu.get(i)-inicisCantContinu.get(i)) > mitjanaAmplitudCants*0.125) {
            indexInsercio = trobaPosicioInsercioOrdenada(llistaCants, new Pair(inicisCantContinu.get(i)/4.0, finalsCantContinu.get(i)/4.0));
            llistaCants.add(indexInsercio, new Pair(inicisCantContinu.get(i)/4.0, finalsCantContinu.get(i)/4.0));
          }
        }
      }

      //Ajusta els cants al seu inici i final real d'una manera més precisa (encara que ja s'hagi fet si té una desviació estàndard elevada, tant els cants com la mitjana d'amplitud poden haver canviat).
      return ajustaCants(llistaCants, amplitudsModificadesAbsolutesReduides);
    }

    /**
      @pre ---
      @post Troba la desviació estàndard, en tant per u, de la durada dels cants de la llista entrada.
      @return La desviació estàndard, en tant per u, de la durada dels cants de la llista entrada
      @param llistaCants Llista d'on extraiem l'inici i el final dels cants per calcular la desviació estàndard de la seva durada
    */
    private static double trobaDesviacioEstandard(ArrayList<Pair<Double, Double> > llistaCants) {
      if (llistaCants.size() > 0) {
        double mitjanaDuradaCant = 0;

        for (Pair<Double, Double> llistaCant : llistaCants) {
          mitjanaDuradaCant += llistaCant.getSecond() - llistaCant.getFirst();
        }

        mitjanaDuradaCant /= llistaCants.size();

        double desviacioEstandard = 0;

        for (Pair<Double, Double> llistaCant : llistaCants) {
          desviacioEstandard += Math.pow(llistaCant.getSecond() - llistaCant.getFirst() - mitjanaDuradaCant, 2);
        }

        desviacioEstandard /= llistaCants.size();
        desviacioEstandard = Math.sqrt(desviacioEstandard);

        //En tant per u
        desviacioEstandard /= mitjanaDuradaCant;

        return desviacioEstandard;
      }
      else return 0;
    }

    /**
```

```
    @pre ---
    @post Troba la desviació estàndard, en tant per u, de la durada dels cants de la llista entrada.
    @return La desviació estàndard, en tant per u, de la durada dels cants de la llista entrada
    @param llistaCants Llista d'on extraiem les durades dels cants per calcular-ne la desviació estàndard
  */
  private static double trobaDesviacioEstandardValors(ArrayList<Double> llistaValors) {
    if (llistaValors.size() > 0) {
      double mitjanaValor = 0;

      for (Double llistaValor : llistaValors) {
        mitjanaValor += llistaValor;
      }

      mitjanaValor /= llistaValors.size();

      double desviacioEstandard = 0;

      for (Double llistaValor : llistaValors) {
        desviacioEstandard += Math.pow(llistaValor - mitjanaValor, 2);
      }

      desviacioEstandard /= llistaValors.size();
      desviacioEstandard = Math.sqrt(desviacioEstandard);

      //En tant per u
      desviacioEstandard /= mitjanaValor;

      return desviacioEstandard;
    }
    else return 0;
  }

  /**
    @pre ---
    @post Retorna la llista de cants entrada després d'haver agrupat aquells cants quasi contigus que siguin considerats una mateixa iteració de cant amb una petita baixada d'amplitud en lloc de dos cants diferents.
    @return La llista de cants entrada després d'haver agrupat aquells cants quasi contigus que siguin considerats una mateixa iteració de cant amb una petita baixada d'amplitud en lloc de dos cants diferents
    @param llistaCants Llista d'on extraiem l'inici i el final de cada cant
  */
  private static ArrayList<Pair<Double, Double> > agrupaCants(ArrayList<Pair<Double, Double> > llistaCants) {
    for(int i = 0; i < llistaCants.size() - 1; i++) {
      //Si els cants estan junts (i no son cants llarg i curt enllaçats d'una Tettigettula pygmea)
      if (llistaCants.get(i+1).getFirst() - llistaCants.get(i).getSecond() < 20 || (llistaCants.get(i+1).getFirst() - llistaCants.get(i).getSecond() < 35 && (llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst() > (llistaCants.get(i+1).getSecond() - llistaCants.get(i+1).getFirst())*3 || llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst() < (llistaCants.get(i+1).getSecond() - llistaCants.get(i+1).getFirst())/3) && !(llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst() < 525 && llistaCants.get(i).getSecond() - llistaCants.get(i).getFirst() > 375 && llistaCants.get(i+1).getSecond() - llistaCants.get(i+1).getFirst() < 80 && llistaCants.get(i+1).getSecond() - llistaCants.get(i+1).getFirst() > 42))) {
        llistaCants.set(i, new Pair(llistaCants.get(i).getFirst(), llistaCants.get(i+1).getSecond()));
        llistaCants.remove(i+1);
        i--;
      }
    }
    return llistaCants;
  }

  /**
    @pre ---
    @post Retorna la posició, dins de la llista llistaCants, on es pot inserir el cant cant i mantenir la llista ordenada cronològicament.
    @return La posició, dins de la llista llistaCants, on es pot inserir el cant cant i mantenir la llista ordenada cronològicament
    @param llistaCants Llista d'on extraiem l'inici i el final de cada cant i on hem d'inserir el cant cant
    @param cant Parella que ens indica l'inici i el final del cant que volem inserir a la llista de cants
  */
  private static int trobaPosicioInsercioOrdenada(ArrayList<Pair<Double, Double> > llistaCants, Pair<Double, Double> cant) {
    int i = 0;
    while (i < llistaCants.size() && llistaCants.get(i).getFirst() < cant.getFirst()) {
      i++;
    }
    return i;
  }

  /**
    @pre ---
    @post Retorna les freqüències, en tant per u respecte a la freqüència de mostreig, que suposaran els llindars mínim i màxim per filtrar el so.
    @return Les freqüències, en tant per u respecte a la freqüència de mostreig, que suposaran els llindars mínim i màxim per filtrar el so
    @param llistaAmplFreq Llista d'amplituds respecte la freqüència del fitxer de so
    @param fs Freqüència de mostreig utilitzada a la gravació
  */
  private static Pair<Double, Double> trobaFrequenciesTallAbansDeReduir(ArrayList<Double> llistaAmplFreq, int fs) {
    double freqTall = 2*2000.0/fs; //No vull sons de menys de 2000 Hz.
    int fragment = 1;
    int nValorsFreq = llistaAmplFreq.size();
    int fr = (int) Math.round(freqTall*nValorsFreq);

    double mitjanaFreq = trobaMitjana(fr,fragment, llistaAmplFreq);
    double mitjanaTotal = trobaMitjana(fr,nValorsFreq - fr, llistaAmplFreq);

    while (fr + fragment < nValorsFreq && mitjanaFreq < mitjanaTotal/1.8) {
      fr++;
      mitjanaFreq = trobaMitjana(fr,fragment, llistaAmplFreq);
    }

    if (1.0*fr/(nValorsFreq) > freqTall) freqTall = arrodoneixADivisio(1.0*fr/(nValorsFreq), 200);
    else freqTall = arrodoneixADivisio(freqTall, 200);

    //En cas que hi hagi dos pics de freqüència corresponents a dues espècies de cigala diferents (és habitual que hi hagi una Cicada orni cantant mentre se n'intenta gravar una altra), es queda el cant de l'espècie que no és Cicada orni, ja que se sobreentén que no és la que s'intenta gravar perquè, en cas que se'n sentin dues amb una potència semblant, la Cicada orni serà força més lluny que l'altra, atès que les Cicada tenen el cant més potent amb diferència.
    int iniciPrimerMaxim = 0;
    int iniciSegonMaxim = 0;
    int iniciMinim = 0;
    double potenciaPrimerMaxim = 0;
    double potenciaSegonMaxim = 0;
    double potenciaMinim = Double.MAX_VALUE;
    double potenciaPrimerMaximAux, potenciaSegonMaximAux, potenciaMinimAux;


    int iniciRang = (int)Math.round(freqTall*nValorsFreq);
    int nValorsPeriode = (int)Math.round(2000.0*nValorsFreq*2/fs);
    int nValorsPeriodeMinim = (int)Math.round(1500.0*nValorsFreq*2/fs);
    int limitFinalRang1 = (int)Math.round(7500.0*nValorsFreq*2/fs);
    int limitIniciRang2 = (int)Math.round(7500.0*nValorsFreq*2/fs);
    int limitIniciRangMinim = (int)Math.round(5000.0*nValorsFreq*2/fs);
    int limitFinalRangMinim = (int)Math.round(10000.0*nValorsFreq*2/fs);

    if (iniciRang + nValorsPeriode*3 < nValorsFreq) {
      while (iniciRang + nValorsPeriode < limitFinalRang1) {
        potenciaPrimerMaximAux = 0;
        for(int i = iniciRang; i < iniciRang + nValorsPeriode; i++) {
          potenciaPrimerMaximAux += llistaAmplFreq.get(i);
        }
        if (potenciaPrimerMaximAux > potenciaPrimerMaxim) {
          potenciaPrimerMaxim = potenciaPrimerMaximAux;
          iniciPrimerMaxim = (int)Math.round(iniciRang*fs*1.0/(2.0*nValorsFreq));
        }
        iniciRang++;
      }

      iniciRang = limitIniciRangMinim;

      while (iniciRang + nValorsPeriodeMinim < limitFinalRangMinim) {
        potenciaMinimAux = 0;
        for(int i = iniciRang; i < iniciRang + nValorsPeriodeMinim; i++) {
          potenciaMinimAux += llistaAmplFreq.get(i);
        }
        if (potenciaMinimAux < potenciaMinim) {
          potenciaMinim = potenciaMinimAux;
          iniciMinim = (int)Math.round(iniciRang*fs*1.0/(2.0*nValorsFreq));
        }
        iniciRang++;
      }

      iniciRang = limitIniciRang2;

      while (iniciRang + nValorsPeriode < nValorsFreq) {
        potenciaSegonMaximAux = 0;
        for(int i = iniciRang; i < iniciRang + nValorsPeriode; i++) {
          potenciaSegonMaximAux += llistaAmplFreq.get(i);
        }
        if (potenciaSegonMaximAux > potenciaSegonMaxim) {
          potenciaSegonMaxim = potenciaSegonMaximAux;
          iniciSegonMaxim = (int)Math.round(iniciRang*fs*1.0/(2.0*nValorsFreq));
        }
        iniciRang++;
      }
```

```
        potenciaPrimerMaxim *= nValorsPeriodeMinim*1.0 / nValorsPeriode;
        potenciaSegonMaxim *= nValorsPeriodeMinim*1.0 / nValorsPeriode;

        if (potenciaMinim < potenciaPrimerMaxim/1.5 && potenciaMinim < potenciaSegonMaxim/1.5 && iniciMinim >= iniciPrimerMaxim+nValorsPeriode && iniciMinim+nValorsPeriodeMinim <= iniciSegonMaxim) {
          freqTall = 2*(iniciMinim+2000.0)/fs;
          fragment = 1;
          fr = (int) Math.round(freqTall*nValorsFreq);

          mitjanaFreq = trobaMitjana(fr,fragment, llistaAmplFreq);
          mitjanaTotal = trobaMitjana(fr,nValorsFreq - fr, llistaAmplFreq);

          while (fr + fragment < nValorsFreq && mitjanaFreq < mitjanaTotal/1.8) {
            fr++;
            mitjanaFreq = trobaMitjana(fr,fragment, llistaAmplFreq);
          }

          if (1.0*fr/(nValorsFreq) > freqTall) freqTall = arrodoneixADivisio(1.0*fr/(nValorsFreq), 200);
          else freqTall = arrodoneixADivisio(freqTall, 200);
        }
      }

      fr = (llistaAmplFreq.size())-1;
      mitjanaFreq = trobaMitjana(fr-fragment,fragment, llistaAmplFreq);

      while (fr - fragment >= 0 && mitjanaFreq < mitjanaTotal/2.0) {
        fr--;
        mitjanaFreq = trobaMitjana(fr-fragment,fragment, llistaAmplFreq);
      }

      double freqTallSuperior = arrodoneixADivisio(1.0*fr/(llistaAmplFreq.size()), 200);

      double freqTallReal = freqTall;
      double freqTallSuperiorReal = freqTallSuperior;

      return new Pair(freqTallReal, freqTallSuperiorReal);
    }

    /**
      @pre ---
      @post Retorna la llista d'amplituds respecte al temps entrada després de filtrar-hi el soroll.
      @return La llista d'amplituds respecte al temps entrada després de filtrar-hi el soroll
      @param valorsAmplitudMono Llista d'amplituds respecte al temps
      @param freqTall Llindar mínim de freqüència que considero acceptable
      @param freqTallSuperior Llindar màxim de freqüència que considero acceptable
    */
    private static ArrayList<Double> filtraPerBandPass(ArrayList<Integer> valorsAmplitudMono, double freqTall, double freqTallSuperior) {
      double coefX = Math.min(0.990, Math.max(0.005, arrodoneixADivisio(freqTall, 200))); //Coeficients entre 0 i 1 (no inclosos)
      double coefY = Math.min(0.995, Math.max(0.005, arrodoneixADivisio(freqTallSuperior, 200)));

      int p, q;
      double sum;
      ArrayList<Double> amplitudsModificades = new ArrayList<>();
      for(Integer ampl : valorsAmplitudMono) {
        amplitudsModificades.add(ampl*1.0);
      }

      Pair<ArrayList<Double>, ArrayList<Double> > coeficientsBandPass = obtinguesCoeficientsBandPass(coefX,coefY);
      ArrayList<Double> a = coeficientsBandPass.getFirst();
      ArrayList<Double> b = coeficientsBandPass.getSecond();

      for(int i = a.size(); i < valorsAmplitudMono.size() - 1; i++) {
        sum = 0;
        for(p = 1; p < a.size(); p++) {
          sum -= a.get(p) * amplitudsModificades.get(i-p);
        }
        for(q = 0; q < b.size(); q++) {
          sum += b.get(q) * valorsAmplitudMono.get(i-q);
        }
        amplitudsModificades.set(i, sum);
      }

      return amplitudsModificades;
    }

    /**
      @pre ---
      @post Retorna la freqüència mitjana, sense tenir en compte les que quedin força dels llindars entrats, a partir de la llista d'amplituds respecte a la freqüència entrada.
      @return La freqüència mitjana, sense tenir en compte les que quedin força dels llindars entrats, a partir de la llista d'amplituds respecte a la freqüència entrada
      @param llistaAmplFreq Llista d'amplituds respecte a la freqüència
      @param fs Freqüència de mostreig utilitzada
      @param freqTall Llindar mínim de freqüència que considero acceptable
      @param freqTallSuperior Llindar màxim de freqüència que considero acceptable
    */
    private static double trobaMitjanaFrequencia(ArrayList<Double> llistaAmplFreq, int fs, double freqTall, double freqTallSuperior) {
      double comptador = 0;
      double mitjana = 0;
      double coefAug = 0.5*fs/llistaAmplFreq.size();

      int inici = 0;
      int fi = llistaAmplFreq.size()-1;

      while (inici < llistaAmplFreq.size() && 1.0*inici/llistaAmplFreq.size() < freqTall) inici++;
      while (fi >= 0 && 1.0*fi/llistaAmplFreq.size() > freqTallSuperior) fi--;

      for(int i = inici; i <= fi; i++) {
        mitjana += llistaAmplFreq.get(i)*coefAug*i;
        comptador += llistaAmplFreq.get(i);
      }

      return mitjana / comptador;
    }

    /**
      @pre ---
      @post Troba la desviació estàndard de les freqüències del cant.
      @return La desviació estàndard de les freqüències del cant
      @param llistaAmplFreq Llista d'amplituds respecte a la freqüència
      @param fs Freqüència de mostreig utilitzada
      @param freqTall Llindar mínim de freqüència que considero acceptable
      @param freqTallSuperior Llindar màxim de freqüència que considero acceptable
    */
    private static double trobaSDFrequencia(ArrayList<Double> llistaAmplFreq, int fs, double freqTall, double freqTallSuperior) {
      double coefAug = 0.5*fs/llistaAmplFreq.size();

      int inici = 0;
      int fi = llistaAmplFreq.size()-1;

      while (inici < llistaAmplFreq.size() && 1.0*inici/llistaAmplFreq.size() < freqTall) inici++;
      while (fi >= 0 && 1.0*fi/llistaAmplFreq.size() > freqTallSuperior) fi--;

      ArrayList<Double> llistaPerTrobarSD = new ArrayList<>();
      int nInsercions;
      for(int i = inici; i <= fi; i++) {
        //Redueix el número perquè sinó serien massa elements per inserir.
        nInsercions = (int) Math.round(llistaAmplFreq.get(i) / 100000);
        for(int j = 0; j < nInsercions; j++) {
          llistaPerTrobarSD.add(i * coefAug);
        }
      }

      return trobaDesviacioEstandardValors(llistaPerTrobarSD);
    }

    /**
      @pre ---
      @post Calcula el coeficient de correlació de Pearson entre la llargada dels cants i la dels silencis.
      @return El coeficient de correlació de Pearson entre la llargada dels cants i la dels silencis
      @param llistaCantsDetallada Llista d'inicis i finals de cants concrets
      @param amplitudsModificadesMoltReduides Llista d'amplituds respecte al temps filtrades i escalades
    */
    private static double calculaCorrelacioLlargadaCantsISilencis(ArrayList<Pair<Double, Double>> llistaCantsDetallada, ArrayList<Double> amplitudsModificadesMoltReduides) {
      double mitjanaCants = 0.0;
      double mitjanaSilencis = 0.0;
      double sumaQuadratsCants = 0.0;
      double sumaQuadratsSilencis = 0.0;
      double sumaCantsPerSilencis = 0.0;
      double desviacioCants, desviacioSilencis;
      double duradaCantAux, duradaSilenciAux;
      double mitjanaAmplitudCantAux;
      int nCantsCurts = 0;
      int iterador = 0;
      boolean cantCurt;

      if (llistaCantsDetallada.size() <= 1) return 0.0;
      else {    //Coeficient de correlació de Pearson
        duradaCantAux = llistaCantsDetallada.get(iterador).getSecond() - llistaCantsDetallada.get(iterador).getFirst();
        cantCurt = duradaCantAux <= 700;
        iterador++;
        while (!cantCurt && iterador < llistaCantsDetallada.size() - 1) {
          duradaCantAux = llistaCantsDetallada.get(iterador).getSecond() - llistaCantsDetallada.get(iterador).getFirst();
```

```
            cantCurt = duradaCantAux <= 700;
            iterador++;
        }

        //Només calcula la correlació si es tracta d'un cant curt.
        if (cantCurt) {

            mitjanaAmplitudCantAux = trobaMitjana((int) Math.round(llistaCantsDetallada.get(0).getFirst() * 4), (int) Math.round(duradaCantAux * 4), amplitudsModificadesMoltReduides) * duradaCantAux / 1000;
            mitjanaCants += mitjanaAmplitudCantAux;
            sumaQuadratsCants += Math.pow(mitjanaAmplitudCantAux, 2);

            for(int i = iterador; i < llistaCantsDetallada.size() - 1; i++) {
              if (cantCurt) {
                duradaSilenciAux = llistaCantsDetallada.get(i).getFirst() - llistaCantsDetallada.get(i-1).getSecond();
                mitjanaSilencis += duradaSilenciAux;
                sumaQuadratsSilencis += Math.pow(duradaSilenciAux, 2);
                sumaCantsPerSilencis += mitjanaAmplitudCantAux * duradaSilenciAux;
                nCantsCurts++;
              }
              duradaCantAux = llistaCantsDetallada.get(i).getSecond() - llistaCantsDetallada.get(i).getFirst();
              cantCurt = duradaCantAux <= 700;
              if (cantCurt) {
                mitjanaAmplitudCantAux = trobaMitjana((int) Math.round(llistaCantsDetallada.get(i).getFirst() * 4), (int) Math.round(duradaCantAux * 4), amplitudsModificadesMoltReduides) * duradaCantAux / 1000;
                mitjanaCants += mitjanaAmplitudCantAux;
                sumaQuadratsCants += Math.pow(mitjanaAmplitudCantAux, 2);
              }
            }

            if (cantCurt) {
              duradaSilenciAux = llistaCantsDetallada.get(llistaCantsDetallada.size()-1).getFirst() - llistaCantsDetallada.get(llistaCantsDetallada.size()-2).getSecond();
              mitjanaSilencis += duradaSilenciAux;
              sumaQuadratsSilencis += Math.pow(duradaSilenciAux, 2);
              sumaCantsPerSilencis += mitjanaAmplitudCantAux * duradaSilenciAux;
              nCantsCurts++;
            }

            if (nCantsCurts > 0) {
              mitjanaCants /= (nCantsCurts);
              mitjanaSilencis /= (nCantsCurts);

              desviacioCants = Math.sqrt(sumaQuadratsCants/nCantsCurts - Math.pow(mitjanaCants, 2));
              desviacioSilencis = Math.sqrt(sumaQuadratsSilencis/nCantsCurts - Math.pow(mitjanaSilencis, 2));

              if (desviacioCants > 0 && desviacioSilencis > 0)
                return ((sumaCantsPerSilencis / nCantsCurts - mitjanaCants * mitjanaSilencis) / (desviacioCants * desviacioSilencis));
              else return 0.0;
            }
            else return 0.0;
        }
        else return 0.0;
      }
    }

    /**
      @pre ---
      @post Determina si els cants que hi ha a les posicions posicio-1 i posicio són un cant llarg (el primer) i un cant curt (el segon) enllaçats.
      @return Booleà que indica si els cants que hi ha a les posicions posicio-1 i posicio són un cant llarg (el primer) i un cant curt (el segon) enllaçats
      @param llistaCantsDetallada Llista d'inicis i finals de cants concrets
      @param posicio Enter que ens indica per a quina parella de cants de la llista llistaCantsDetallada hem de comprovar si són un cant llarg (el primer) i un cant curt (el segon) enllaçats
     */
    private static boolean sonCantLlargICantCurt(ArrayList<Pair<Double, Double> > llistaCantsDetallada, int posicio) {

      if (posicio > 0 && llistaCantsDetallada.size() > 1) {
        double distanciaAuxAnterior = llistaCantsDetallada.get(posicio - 1).getSecond() - llistaCantsDetallada.get(posicio - 1).getFirst();
        double distanciaAux = llistaCantsDetallada.get(posicio).getSecond() - llistaCantsDetallada.get(posicio).getFirst();
        double distanciaSilenci = llistaCantsDetallada.get(posicio).getFirst() - llistaCantsDetallada.get(posicio-1).getSecond();
        double distanciaSilenciPosterior;
        if (posicio < llistaCantsDetallada.size() - 1) distanciaSilenciPosterior = llistaCantsDetallada.get(posicio+1).getFirst() - llistaCantsDetallada.get(posicio).getSecond();
        else distanciaSilenciPosterior = -1;

        return distanciaAux > 20 && distanciaAux < 130 && distanciaAuxAnterior > 320 && distanciaAuxAnterior < 12000 && distanciaAuxAnterior / distanciaAux > 3.25 && distanciaSilenci > 20 && distanciaSilenci < 150 && (distanciaSilenciPosterior == -1 || (distanciaSilenciPosterior > 140 && distanciaSilenci < distanciaSilenciPosterior / 2.0));
      }
      else return false;
    }

    /**
      @pre ---
      @post Retorna la semblança entre el cant analitzat i el de l'espècie especie, així com el nivell de confiança global en els valors obtinguts per als paràmetres analitzats.
      @return La semblança entre el cant analitzat i el de l'espècie especie, així com el nivell de confiança global en els valors obtinguts per als paràmetres analitzats
      @param especie Espècie amb la qual hem de determinar la semblança
      @param nivellConfiancaSonCantsCurts Nivell de confiança que els cants siguin curts
      @param nivellConfiancaSonCantsLlargs Nivell de confiança que els cants siguin llargs
      @param nivellConfiancaEsCantLlargIrregular Nivell de confiança que cants siguin llargs i amb freqüències irregulars
      @param duradaMitjanaCantCurt Durada mitjana dels cants curts
      @param duradaMitjanaSilenci Durada mitjana dels silencis entre els cants
      @param percentatgeCantsCurts Percentatge de cants curts sobre el total de cants
      @param nivellConfiancaCantsCurtsAmbCantLlargPrevi Nivell de confiança que els cants són llargs amb cants curts enllaçats
      @param duradaMitjanaCantCurtAmbCantLlargPrevi Durada mitjana dels cants curts enllaçats a un cant llarg
      @param duradaMitjanaCantLlargAmbCantCurtPosterior Durada mitjana dels cants llargs amb un cant curt enllaçat
      @param duradaMitjanaSilenciEntreCantLlargICantCurtPosterior Durada mitjana dels silencis entre els cants llarg i curt enllaçats
      @param pendentPrimeraMeitatCantsCurts Pendent de l'amplitud de la primera meitat dels cants curts
      @param pendentSegonaMeitatCantsCurts Pendent de l'amplitud de la segon meitat dels cants curts
      @param pendentPrimeraMeitatCantsLlargs Pendent de l'amplitud de la primera meitat dels cants llargs
      @param pendentSegonaMeitatCantsLlargs Pendent de l'amplitud de la segona meitat dels cants llargs
      @param correlacioPotenciaCantsILlargadaSilencis Correlació entre la potència dels cants i la llargada dels silencis contigus
      @param frequenciaMaximaMostreig Freqüència de mostreig real
      @param frequenciaMitjana Freqüència mitjana del acnt
      @param pic1Frequencia Primer pic de freqüència del cant
      @param pic2Frequencia Segon pic de freqüència del cant
      @param pic3Frequencia Tercer pic de freqüència del cant
      @param nombreGrupsDePolsosPerSegon Nombre de grups de polsos per segon als cants llargs
      @param nivellConfiancaNombreGrupsDePolsosPerSegon Nivell confiança en el nombre de grups de polsos per segon als cants llargs
      @param definicioGrupsDePolsos Definició dels grups de polsos (com més definició, més fàcilment distingibles són)
      @param tempsDurada Durada de la gravació
      @param proporcioSilenci Proporció de temps de silenci respecte al temps total de la gravació
      @param nBatecsAlesOCantsIncorrectes Nombre de batecs d'ales o curts increments sobtats d'amplitud interpretats com a cant
      @param nCants Nombre total d'iteracions de cant a l'enregistrament
      @param mitjanaAmplitud Amplitud mitjana del fitxer de so
      @param mitjanaAmplitudCant Amplitud mitjana del cant continguts al fitxer de so
     */
    private static Pair<Double, Double> trobaSemblancaAmbCigala(
        String especie,
        double nivellConfiancaSonCantsCurts,
        double nivellConfiancaSonCantsLlargs,
        double nivellConfiancaEsCantLlargIrregular,
        double duradaMitjanaCantCurt,
        double duradaMitjanaSilenci,
        double percentatgeCantsCurts,
        double nivellConfiancaCantsCurtsAmbCantLlargPrevi,
        double duradaMitjanaCantCurtAmbCantLlargPrevi,
        double duradaMitjanaCantLlargAmbCantCurtPosterior,
        double duradaMitjanaSilenciEntreCantLlargICantCurtPosterior,
        double pendentPrimeraMeitatCantsCurts,
        double pendentSegonaMeitatCantsCurts,
        double pendentPrimeraMeitatCantsLlargs,
        double pendentSegonaMeitatCantsLlargs,
        double correlacioPotenciaCantsILlargadaSilencis,
        double frequenciaMaximaMostreig,
        double frequenciaMitjana,
        int pic1Frequencia,
        int pic2Frequencia,
        int pic3Frequencia,
        double nombreGrupsDePolsosPerSegon,
        double nivellConfiancaNombreGrupsDePolsosPerSegon,
        double definicioGrupsDePolsos,
        double tempsDurada,
        double proporcioSilenci,
        int nBatecsAlesOCantsIncorrectes,
```

```
    double nCants,
    double mitjanaAmplitud,
    double mitjanaAmplitudCant) {

    double potencia = 0.65;

      if (especie.compareTo("Cicada orni") == 0) {
        /*
        //Cicada orni
        Criteris:
                                        Importància base    0%      100%    100%    0%
        Mitjana durada cant curt                -0,75       25      50      170     400
        Nivell confiança són cants curts        -1          0       0,75    -
        Mitjana durada silenci                  -0,75       25      50      170     400
        Nivell confiança cants curts amb cant llarg previ       -1      0,4     0,1     0       -
        Nivell confiança cant llarg irregular   -0,5        0,5     0
        Freqüència mitjana                                      -1      3000    4500    6500    8000
        Nivell confiança grups de polsos per segon
          -0,25         1       0                       -
        Math.min(1, 4.0*batecs ales / tempsDurada)
          -0,2          1       0                       -
        Proporció silenci                       -0,25       0       0,3     0,7     1


        Mínima frequenciaMaximaMostreig = 7000
        Confiança freqüència = (4-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-7000)/17000),5))/4
        freqüència += 1600*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-7000)/17000),5)
        */

        //Inicialitza la semblança a 1 i els valors que no encaixin amb els de Cicada orni aniran decrementant aquest valor.
        double semblancaCicadaOrni = 1.0;

        //Importància que dóna a cada paràmetre que té en compte
        double importanciaBaseMitjanaDuradaCantCurt = 0.75;
        double importanciaBaseSonCantsCurts = 1.0;
        double importanciaBaseMitjanaDuradaSilencis = 0.75;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
        double importanciaBaseCantLlargIrregular = 0.5;
        double importanciaBaseMitjanaFrequencia = 1.0;
        double importanciaBaseNivellConfiancaNGP = 0.25;
        double importanciaBaseBatecsAles = 0.2;
        double importanciaBaseProporcioSilenci = 0.25;

        frequenciaMitjana += 1600*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-7000)/17000),5);

        //Determina si la freqüència de mostreig ens permet comprovar si es tracta d'una Cicada orni.
        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 7000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;

        //Determina, per a cada valor, en quina mesura ha de decrementar la semblança amb una Cicada orni.
        double valorMitjanaDuradaCantCurt = Math.pow(((duradaMitjanaCantCurt < 50) ? Math.max(0, 1.0*(duradaMitjanaCantCurt - 25) / (50-25)) : ((duradaMitjanaCantCurt > 170) ? Math.max(0, 1 - 1.0*(duradaMitjanaCantCurt - 170) / (400-170)) : 1)), potencia);
        double valorSonCantsCurts = Math.pow(Math.min(1, nivellConfiancaSonCantsCurts / 0.75), potencia);
        double valorMitjanaDuradaSilencis = Math.pow(((duradaMitjanaSilenci < 50) ? Math.max(0, 1.0*(duradaMitjanaSilenci - 25) / (50-25)) : ((duradaMitjanaSilenci > 170) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenci - 170) / (400-170)) : 1)), potencia);
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorEsCantLlargIrregular = Math.min(1, 1 - nivellConfiancaEsCantLlargIrregular + 0.5);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 4500) ? Math.max(0, 1.0*(frequenciaMitjana - 3000) / (4500-3000)) : ((frequenciaMitjana > 6500) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 6500) / (8000-6500)) : 1)), potencia);
        double valorNivellConfiancaNGP = 1 - nivellConfiancaNombreGrupsDePolsosPerSegon;
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(((proporcioSilenci < 0.3) ? Math.max(0, 1.0*(proporcioSilenci - 0) / (0.3-0)) : ((proporcioSilenci > 0.7) ? Math.max(0, 1 - 1.0*(proporcioSilenci - 0.7) / (1-0.7)) : 1)), potencia);

        //Decrementa la semblança amb una Cicada orni a partir de les diferències trobades entre els valors del cant analitzat i els d'aquesta espècie.
        semblancaCicadaOrni -= (1 - valorMitjanaDuradaCantCurt) * importanciaBaseMitjanaDuradaCantCurt;
        semblancaCicadaOrni -= (1 - valorSonCantsCurts) * importanciaBaseSonCantsCurts;
        semblancaCicadaOrni -= (1 - valorMitjanaDuradaSilencis) * importanciaBaseMitjanaDuradaSilencis;
        semblancaCicadaOrni -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        semblancaCicadaOrni -= (1 - valorEsCantLlargIrregular) * importanciaBaseCantLlargIrregular;
        if (fiabilitatIdentificacio != -1) semblancaCicadaOrni -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        semblancaCicadaOrni -= (1 - valorNivellConfiancaNGP) * importanciaBaseNivellConfiancaNGP;
        semblancaCicadaOrni -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaCicadaOrni -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaCicadaOrni = Math.max(0,semblancaCicadaOrni);
        return new Pair(semblancaCicadaOrni, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Cicada barbara lusitanica") == 0) {
        /*
        //Cicada barbara lusitanica
        Criteris:
                                        Importància base    0%      100%    100%    0%
        Nivell confiança són cants llargs       -1          0       0,75    -
        Nivell confiança cants curts amb cant llarg previ       -0,75       0,4     0,1     0       -
        Pendent primera meitat cant llarg       -0,05       0,75    0,9     1,25    1,6
        Pendent segona meitat cant llarg        -0,05       0,7     0,95    1,1     1,4
        Freqüència mitjana                      -1          5250    6250    7250    8250
        Nivell confiança grups de polsos per segon      -0,1        0       1       -
        Nombre grups de polsos != corsica fairmairei    -0,25       59 <= NGP <= 65     NGP < 59 || NGP > 65    -
        Nombre grups de polsos != garricola     -0,225      66 <= NGP <= 71     NGP < 66 || NGP > 71    -
        Nombre grups de polsos != haematodes    -0,175      90 <= NGP <= 105    NGP < 90 || NGP > 105   -
        Nombre grups de polsos != quadrisignata     -0,2        73 <= NGP <= 80     NGP < 73 || NGP > 80    -
        Nombre grups de polsos != tomentosa     -0,15       145 <= NGP <= 180   NGP < 145 || NGP > 180  -
        Proporció silenci                       -0,15       0,5     0,2     0       -

        Mínima frequenciaMaximaMostreig = 9000
        Confiança freqüència = (4-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),5))/4
        freqüència += 1600*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),5)
        */

        double semblancaCicadaBarbaraLusitanica = 1.0;

        double importanciaBaseSonCantsLlargs = 1.0;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 0.75;
        double importanciaBasePendentPrimeraMeitatCantLlarg = 0.05;
        double importanciaBasePendentSegonaMeitatCantLlarg = 0.05;
        double importanciaBaseMitjanaFrequencia = 1.0;
        double importanciaBaseNivellConfiancaNGP = 0.1;
        double importanciaBaseNGPDiferentCorsicaFairmairei = 0.25;
        double importanciaBaseNGPDiferentGarricola = 0.225;
        double importanciaBaseNGPDiferentHaematodes = 0.175;
        double importanciaBaseNGPDiferentQuadrisignata = 0.2;
        double importanciaBaseNGPDiferentTomentosa = 0.15;
        double importanciaBaseProporcioSilenci = 0.15;

        frequenciaMitjana += 1600*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),5);

        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 9000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;

        double valorSonCantsLlargs = Math.pow(Math.min(1, Math.max(nivellConfiancaSonCantsLlargs, (nivellConfiancaEsCantLlargIrregular > 0.7) ? nivellConfiancaEsCantLlargIrregular*0.75 : 0.0) / 0.75), potencia);
```

```
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorPendentPrimeraMeitatCantLlarg = Math.pow(((pendentPrimeraMeitatCantsLlargs < 0.9) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsLlargs - 0.75) / (0.9-0.75)) : ((pendentPrimeraMeitatCantsLlargs > 1.25) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsLlargs - 1.25) / (1.6-1.25)) : 1)), potencia);
        double valorPendentSegonaMeitatCantLlarg = Math.pow(((pendentSegonaMeitatCantsLlargs < 0.95) ? Math.max(0, 1.0*(pendentSegonaMeitatCantsLlargs - 0.7) / (0.95-0.7)) : ((pendentSegonaMeitatCantsLlargs > 1.1) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsLlargs - 1.1) / (1.4-1.1)) : 1)), potencia);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 6250) ? Math.max(0, 1.0*(frequenciaMitjana - 5250) / (6250-5250)) : ((frequenciaMitjana > 7250) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 7250) / (8250-7250)) : 1)), potencia);
        double valorNivellConfiancaNGP = 1 - nivellConfiancaNombreGrupsDePolsosPerSegon;
        double valorNGPDiferentCorsicaFairmairei = (nombreGrupsDePolsosPerSegon >= 59 && nombreGrupsDePolsosPerSegon <= 65) ? 0 : 1;
        double valorNGPDiferentGarricola = (nombreGrupsDePolsosPerSegon >= 66 && nombreGrupsDePolsosPerSegon <= 71) ? 0 : 1;
        double valorNGPDiferentHaematodes = (nombreGrupsDePolsosPerSegon >= 90 && nombreGrupsDePolsosPerSegon <= 105) ? 0 : 1;
        double valorNGPDiferentQuadrisignata = (nombreGrupsDePolsosPerSegon >= 73 && nombreGrupsDePolsosPerSegon <= 80) ? 0 : 1;
        double valorNGPDiferentTomentosa = (nombreGrupsDePolsosPerSegon >= 145 && nombreGrupsDePolsosPerSegon <= 180) ? 0 : 1;
        double valorProporcioSilenci = Math.pow(Math.min(1, Math.max(0, (1 - proporcioSilenci - 0.5) / (0.3))), potencia);


        semblancaCicadaBarbaraLusitanica -= (1 - valorSonCantsLlargs) * importanciaBaseSonCantsLlargs;
        semblancaCicadaBarbaraLusitanica -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        semblancaCicadaBarbaraLusitanica -= (1 - valorPendentPrimeraMeitatCantLlarg) * importanciaBasePendentPrimeraMeitatCantLlarg;
        semblancaCicadaBarbaraLusitanica -= (1 - valorPendentSegonaMeitatCantLlarg) * importanciaBasePendentSegonaMeitatCantLlarg;
        if (fiabilitatIdentificacio != -1) semblancaCicadaBarbaraLusitanica -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        semblancaCicadaBarbaraLusitanica -= (1 - valorNivellConfiancaNGP) * importanciaBaseNivellConfiancaNGP;
        semblancaCicadaBarbaraLusitanica -= (1 - valorNGPDiferentCorsicaFairmairei) * importanciaBaseNGPDiferentCorsicaFairmairei;
        semblancaCicadaBarbaraLusitanica -= (1 - valorNGPDiferentGarricola) * importanciaBaseNGPDiferentGarricola;
        semblancaCicadaBarbaraLusitanica -= (1 - valorNGPDiferentHaematodes) * importanciaBaseNGPDiferentHaematodes;
        semblancaCicadaBarbaraLusitanica -= (1 - valorNGPDiferentQuadrisignata) * importanciaBaseNGPDiferentQuadrisignata;
        semblancaCicadaBarbaraLusitanica -= (1 - valorNGPDiferentTomentosa) * importanciaBaseNGPDiferentTomentosa;
        semblancaCicadaBarbaraLusitanica -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaCicadaBarbaraLusitanica = Math.max(0,semblancaCicadaBarbaraLusitanica);

        //Redueixo la semblança atesa la poca restrictivitat de les condicions que es requereix per a aquesta espècie.
        semblancaCicadaBarbaraLusitanica /= 2.0;
        return new Pair(semblancaCicadaBarbaraLusitanica, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Cicadatra atra") == 0) {
        /*
        //Cicadatra atra
        Criteris:
                                        Importància base    0%      100%    100%    0%
          Mitjana durada cant curt                  -0,6    100     200     -
          Mitjana durada silenci                -0,4    50      100     300     400
          Nivell confiança cants curts amb cant llarg previ           -1      0,4     0,1     0       -
          Pendent primera meitat cant llarg             -0,15   0,85    0,95    1,05    1,2
          Pendent segona meitat cant llarg              -0,15   0,85    0,95    1,05    1,2
          Nivell confiança cant llarg irregular     -0,15   0,5     0
          Freqüència mitjana                -1      9250    10250   12250   13250
          Math.min(2, batecs ales / tempsDurada)
          -0,2    2       0       -
          Nivell confiança grups de polsos per segon
          -0,25   1       0       -
          Nombre grups de polsos != corsica fairmairei          -0,25   59 <= NGP <= 65     NGP < 59 || NGP > 65    -
          Nombre grups de polsos != garricola           -0,225  66 <= NGP <= 71   NGP < 66 || NGP > 71     -
          Nombre grups de polsos != haematodes
          -0,175  90 <= NGP <= 105    NGP < 90 || NGP > 105   -
          Nombre grups de polsos != quadrisignata
          -0,2    73 <= NGP <= 80NGP < 73 || NGP > 80     -
          Nombre grups de polsos != tomentosa           -0,15   145 <= NGP <= 180   NGP < 145 || NGP > 180 -
          Proporció silenci si esCantCurt > 0.75                -0,4    0,15    0,3     0,55    0,75
          Proporció silenci si esCantLlarg > 0.75               -0,4    0,5     0,1     0       -

          Mínima frequenciaMaximaMostreig = 11000
          Confiança freqüència = (4-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),10))/4
          freqüència += 1500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),10)
          */

        double semblancaCicadatraAtra = 1.0;

        double importanciaBaseMitjanaDuradaCantCurt = 0.6;
        double importanciaBaseMitjanaDuradaSilencis = 0.4;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
        double importanciaBasePendentPrimeraMeitatCantLlarg = 0.15;
        double importanciaBasePendentSegonaMeitatCantLlarg = 0.15;
        double importanciaBaseCantLlargIrregular = 0.15;
        double importanciaBaseMitjanaFrequencia = 1.0;
        double importanciaBaseBatecsAles = 0.2;
        double importanciaBaseNivellConfiancaNGP = 0.25;
        double importanciaBaseNGPDiferentCorsicaFairmairei = 0.25;
        double importanciaBaseNGPDiferentGarricola = 0.225;
        double importanciaBaseNGPDiferentHaematodes = 0.175;
        double importanciaBaseNGPDiferentQuadrisignata = 0.2;
        double importanciaBaseNGPDiferentTomentosa = 0.15;
        double importanciaBaseProporcioSilenci = 0.4;


        frequenciaMitjana += 1500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),10);

        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 11000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;


        double proporcioSilenciNomesCantsCurts = duradaMitjanaSilenci / (duradaMitjanaSilenci + duradaMitjanaCantCurt);

        double valorMitjanaDuradaCantCurt = Math.pow(Math.min(1, Math.max(0, (duradaMitjanaCantCurt - 100) / (100))), potencia);
        double valorMitjanaDuradaSilencis = Math.pow(((duradaMitjanaSilenci < 100) ? Math.max(0, 1.0*(duradaMitjanaSilenci - 50) / (100-50)) : ((duradaMitjanaSilenci > 300) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenci - 300) / (500-300)) : 1)), potencia);
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorPrimeraMeitatCantLlarg = Math.pow(((pendentPrimeraMeitatCantsLlargs < 0.95) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsLlargs - 0.85) / (0.95-0.85)) : ((pendentPrimeraMeitatCantsLlargs > 1.05) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsLlargs - 1.05) / (1.2-1.05)) : 1)), potencia);
        double valorSegonaMeitatCantLlarg = Math.pow(((pendentSegonaMeitatCantsLlargs < 0.95) ? Math.max(0, 1.0*(pendentSegonaMeitatCantsLlargs - 0.85) / (0.95-0.85)) : ((pendentSegonaMeitatCantsLlargs > 1.05) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsLlargs - 1.05) / (1.2-1.05)) : 1)), potencia);
        double valorEsCantLlargIrregular = Math.min(1, 1 - nivellConfiancaEsCantLlargIrregular + 0.5);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 10250) ? Math.max(0, 1.0*(frequenciaMitjana - 9250) / (10250-9250)) : ((frequenciaMitjana > 12250) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 12250) / (13250-12250)) : 1)), potencia);
        double valorNivellConfiancaNGP = 1 - nivellConfiancaNombreGrupsDePolsosPerSegon;
        double valorNGPDiferentCorsicaFairmairei = (nombreGrupsDePolsosPerSegon >= 59 && nombreGrupsDePolsosPerSegon <= 65) ? 0 : 1;
        double valorNGPDiferentGarricola = (nombreGrupsDePolsosPerSegon >= 66 && nombreGrupsDePolsosPerSegon <= 71) ? 0 : 1;
        double valorNGPDiferentHaematodes = (nombreGrupsDePolsosPerSegon >= 90 && nombreGrupsDePolsosPerSegon <= 105) ? 0 : 1;
        double valorNGPDiferentQuadrisignata = (nombreGrupsDePolsosPerSegon >= 73 && nombreGrupsDePolsosPerSegon <= 80) ? 0 : 1;
```

```
        double valorNGPDiferentTomentosa = (nombreGrupsDePolsosPerSegon >=
145 && nombreGrupsDePolsosPerSegon <= 180) ? 0 : 1;
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes
/ tempsDurada);
        double valorProporcioSilenci = 1.0;
        if (nivellConfiancaSonCantsCurts > 0.75)
          valorProporcioSilenci = Math.pow(((proporcioSilenciNomesCantsCurts <
0.3) ? Math.max(0, 1.0*(proporcioSilenciNomesCantsCurts - 0.15) / (0.3-0.15)) :
((proporcioSilenciNomesCantsCurts > 0.55) ? Math.max(0, 1 -
1.0*(proporcioSilenciNomesCantsCurts - 0.55) / (0.75-0.55)) : 1)), potencia);
        else if (nivellConfiancaSonCantsLlargs > 0.75)
          valorProporcioSilenci = Math.pow(Math.min(1, Math.max(0, (1 -
proporcioSilenci - 0.5) / (0.4))), potencia);

        if (percentatgeCantsCurts > 0) semblancaCicadatraAtra -= (1 -
valorMitjanaDuradaCantCurt) * importanciaBaseMitjanaDuradaCantCurt;
        if (percentatgeCantsCurts > 0) semblancaCicadatraAtra -= (1 -
valorMitjanaDuradaSilencis) * importanciaBaseMitjanaDuradaSilencis;
        semblancaCicadatraAtra -= (1 - valorCantsCurtsAmbCantLlargPrevi) *
importanciaBaseCantsCurtsAmbCantLlargPrevi;
        if (percentatgeCantsCurts < 100) semblancaCicadatraAtra -= (1 -
valorPrimeraMeitatCantLlarg) * importanciaBasePendentPrimeraMeitatCantLlarg;
        if (percentatgeCantsCurts < 100) semblancaCicadatraAtra -= (1 -
valorSegonaMeitatCantLlarg) * importanciaBasePendentSegonaMeitatCantLlarg;
        semblancaCicadatraAtra -= (1 - valorEsCantLlargIrregular) *
importanciaBaseCantLlargIrregular;
        if (fiabilitatIdentificacio != -1) semblancaCicadatraAtra -= (1 -
valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        if (percentatgeCantsCurts < 100) semblancaCicadatraAtra -= (1 -
valorNivellConfiancaNGP) * importanciaBaseNivellConfiancaNGP;
        semblancaCicadatraAtra -= (1 - valorNGPDiferentCorsicaFairmairei) *
importanciaBaseNGPDiferentCorsicaFairmairei;
        semblancaCicadatraAtra -= (1 - valorNGPDiferentGarricola) *
importanciaBaseNGPDiferentGarricola;
        semblancaCicadatraAtra -= (1 - valorNGPDiferentHaematodes) *
importanciaBaseNGPDiferentHaematodes;
        semblancaCicadatraAtra -= (1 - valorNGPDiferentQuadrisignata) *
importanciaBaseNGPDiferentQuadrisignata;
        semblancaCicadatraAtra -= (1 - valorNGPDiferentTomentosa) *
importanciaBaseNGPDiferentTomentosa;
        semblancaCicadatraAtra -= (2 - valorBatecsAles) *
importanciaBaseBatecsAles;
        semblancaCicadatraAtra -= (1 - valorProporcioSilenci) *
importanciaBaseProporcioSilenci;

        semblancaCicadatraAtra = Math.max(0,semblancaCicadatraAtra);
        return new Pair(semblancaCicadatraAtra, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Cicadetta brevipennis") == 0) {
        /*
        //Cicadetta brevipennis
        Criteris:
                                          Importància base      0%      100%    100%
0%
          Durada mitjana silenci                  -0,2        30      140     2000
           5000
          Nivell confiança cants curts amb cant llarg previ          -1        0
0,3     1       -
          Durada mitjana cant curt amb cant llarg previ              -0,5
0       40      90      130
          Durada mitjana cant llarg amb cant curt posterior          -0,5
700     1800    7000    10000
          Durada silenci entre cant curt i cant llarg
           -0,4         20      50      95      130
          Pendent primera meitat cant llarg          -0,1          0,85
           1,1     1,35          1,7
          Pendent segona meitat cant llarg        -0,1        0,7     0,95
1,1     1,45
          Freqüència mitjana              -0,85       12750   14250
16750       18250
          Math.min(2, batecs ales / tempsDurada)
           -0,2         2       0                  -
          Nombre grups de polsos per segon          -0,4        100     120
170     200
          Nombre grups de polsos != corsica fairmairei          -0,15    59
<= NGP <= 65          NGP < 59 || NGP > 65    -
          Nombre grups de polsos != garricola          -0,125    66 <= NGP <=
71          NGP < 66 || NGP > 71    -
          Nombre grups de polsos != haematodes
           -0,075    90 <= NGP <= 105    NGP < 90 || NGP > 105          -
          Nombre grups de polsos != quadrisignata
           -0,1     73 <= NGP <= 80NGP < 73 || NGP > 80    -
          Proporció silencis              -0,1        0       0,2     0,7
1

          Mínima frequenciaMaximaMostreig = 11000
          Confiança freqüència = (1.5-Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),2.7))/1.5
          freqüència += 7000*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),2.7)
        */
        double semblancaCicadettaBrevipennis = 1.0;

        double importanciaBaseMitjanaDuradaSilencis = 0.2;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1;
        double importanciaBaseMitjanaDuradaCantsCurtsAmbCantLlargPrevi = 0.5;
        double importanciaBaseMitjanaDuradaCantsLlargsAmbCantCurtPosterior =
0.5;
        double importanciaBaseMitjanaDuradaSilencisEntreCantCurtICantLlarg = 0.4;
        double importanciaBasePendentPrimeraMeitatCantLlarg = 0.1;
        double importanciaBasePendentSegonaMeitatCantLlarg = 0.1;
        double importanciaBaseMitjanaFrequencia = 0.85;
        double importanciaBaseBatecsAles = 0.2;
        double importanciaBaseNGPDiferentCorsicaFairmairei = 0.4;
        double importanciaBaseNGPDiferentGarricola = 0.15;
        double importanciaBaseNGPDiferentHaematodes = 0.125;
        double importanciaBaseNGPDiferentQuadrisignata = 0.075;
        double importanciaBaseProporcioSilenci = 0.1;


        frequenciaMitjana += 7000*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),2.7);

        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 11000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200)
fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;



        double valorMitjanaDuradaSilencis = Math.pow(((duradaMitjanaSilenci <
140) ? Math.max(0, 1.0*(duradaMitjanaSilenci - 30) / (140-30)) :
((duradaMitjanaSilenci > 2000) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenci - 2000) /
(5000-2000)) : 1)), potencia);
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1,
Math.max(0, nivellConfiancaCantsCurtsAmbCantLlargPrevi/0.3)), potencia);
        double valorMitjanaDuradaCantsCurtsAmbCantLlargPrevi =
Math.pow(((duradaMitjanaCantCurtAmbCantLlargPrevi < 40) ? Math.max(0,
1.0*(duradaMitjanaCantCurtAmbCantLlargPrevi - 0) / (40-0)) :
((duradaMitjanaCantCurtAmbCantLlargPrevi > 90) ? Math.max(0, 1 -
1.0*(duradaMitjanaCantCurtAmbCantLlargPrevi - 90) / (130-90)) : 1)), potencia);
        double valorMitjanaDuradaCantsLlargsAmbCantCurtPosterior =
Math.pow(((duradaMitjanaCantLlargAmbCantCurtPosterior < 1800) ? Math.max(0,
1.0*(duradaMitjanaCantLlargAmbCantCurtPosterior - 700) / (1800-700)) :
((duradaMitjanaCantLlargAmbCantCurtPosterior > 7000) ? Math.max(0, 1 -
1.0*(duradaMitjanaCantLlargAmbCantCurtPosterior - 7000) / (10000-7000)) : 1)),
potencia);
        double valorMitjanaDuradaSilencisEntreCantCurtICantLlarg =
Math.pow(((duradaMitjanaSilenciEntreCantLlargICantCurtPosterior < 50) ?
Math.max(0, 1.0*(duradaMitjanaSilenciEntreCantLlargICantCurtPosterior - 20) / (50-
20)) : ((duradaMitjanaSilenciEntreCantLlargICantCurtPosterior > 95) ? Math.max(0, 1 -
1.0*(duradaMitjanaSilenciEntreCantLlargICantCurtPosterior - 95) / (130-95)) : 1)),
potencia);
        double valorPendentPrimeraMeitatCantLlarg =
Math.pow(((pendentPrimeraMeitatCantsLlargs < 1.1) ? Math.max(0,
1.0*(pendentPrimeraMeitatCantsLlargs - 0.85) / (1.1-0.85)) :
((pendentPrimeraMeitatCantsLlargs > 1.3) ? Math.max(0, 1 -
1.0*(pendentPrimeraMeitatCantsLlargs - 1.3) / (1.7-1.3)) : 1)), potencia);
        double valorPendentSegonaMeitatCantLlarg =
Math.pow(((pendentSegonaMeitatCantsLlargs < 0.95) ? Math.max(0,
1.0*(pendentSegonaMeitatCantsLlargs - 0.7) / (0.95-0.7)) :
((pendentSegonaMeitatCantsLlargs > 1.1) ? Math.max(0, 1 -
1.0*(pendentSegonaMeitatCantsLlargs - 1.1) / (1.45-1.1)) : 1)), potencia);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 14250) ?
Math.max(0, 1.0*(frequenciaMitjana - 12750) / (14250-12750)) : ((frequenciaMitjana
> 16750) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 16750) / (18250-16750)) : 1)),
potencia);
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes
/ tempsDurada);
        double valorNGPDiferentCorsicaFairmairei =
(nombreGrupsDePolsosPerSegon >= 59 && nombreGrupsDePolsosPerSegon <= 65) ? 0
: 1;
        double valorNGPDiferentGarricola = (nombreGrupsDePolsosPerSegon >= 66
&& nombreGrupsDePolsosPerSegon <= 71) ? 0 : 1;
        double valorNGPDiferentHaematodes = (nombreGrupsDePolsosPerSegon >=
90 && nombreGrupsDePolsosPerSegon <= 105) ? 0 : 1;
        double valorNGPDiferentQuadrisignata = (nombreGrupsDePolsosPerSegon
>= 73 && nombreGrupsDePolsosPerSegon <= 80) ? 0 : 1;
        double valorProporcioSilenci = Math.pow(((proporcioSilenci < 0.2) ?
Math.max(0, 1.0*(proporcioSilenci - 0) / (0.2-0)) : ((proporcioSilenci > 0.7) ?
Math.max(0, 1 - 1.0*(proporcioSilenci - 0.7) / (1-0.7)) : 1)), potencia);


        semblancaCicadettaBrevipennis -= (1 - valorMitjanaDuradaSilencis) *
importanciaBaseMitjanaDuradaSilencis;
        semblancaCicadettaBrevipennis -= (1 - valorCantsCurtsAmbCantLlargPrevi) *
importanciaBaseCantsCurtsAmbCantLlargPrevi;
        semblancaCicadettaBrevipennis -= (1 -
valorMitjanaDuradaCantsCurtsAmbCantLlargPrevi) *
importanciaBaseMitjanaDuradaCantsCurtsAmbCantLlargPrevi;
```

```
        semblancaCicadettaBrevipennis -= (1 - valorMitjanaDuradaCantsLlargsAmbCantCurtPosterior) * importanciaBaseMitjanaDuradaCantsLlargsAmbCantCurtPosterior;
        semblancaCicadettaBrevipennis -= (1 - valorMitjanaDuradaSilencisEntreCantCurtICantLlarg) * importanciaBaseMitjanaDuradaSilencisEntreCantCurtICantLlarg;
        semblancaCicadettaBrevipennis -= (1 - valorPendentPrimeraMeitatCantLlarg) * importanciaBasePendentPrimeraMeitatCantLlarg;
        semblancaCicadettaBrevipennis -= (1 - valorPendentSegonaMeitatCantLlarg) * importanciaBasePendentSegonaMeitatCantLlarg;
        if (fiabilitatIdentificacio != -1) semblancaCicadettaBrevipennis -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        semblancaCicadettaBrevipennis -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaCicadettaBrevipennis -= (1 - valorNGPDiferentCorsicaFairmairei) * importanciaBaseNGPDiferentCorsicaFairmairei;
        semblancaCicadettaBrevipennis -= (1 - valorNGPDiferentGarricola) * importanciaBaseNGPDiferentGarricola;
        semblancaCicadettaBrevipennis -= (1 - valorNGPDiferentHaematodes) * importanciaBaseNGPDiferentHaematodes;
        semblancaCicadettaBrevipennis -= (1 - valorNGPDiferentQuadrisignata) * importanciaBaseNGPDiferentQuadrisignata;
        semblancaCicadettaBrevipennis -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaCicadettaBrevipennis = Math.max(0,semblancaCicadettaBrevipennis);
        return new Pair(semblancaCicadettaBrevipennis, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Cicadetta cerdaniensis") == 0) {
        /*
        //Cicadetta cerdaniensis
        Criteris:
                                    Importància base    0%    100%    100%    0%
          Mitjana durada cant curt                  -0,75    20    30    180    300
          Nivell confiança són cants curts          -1    0,75    1    -
          Mitjana durada silenci                    -0,6    100    170    1500    2000
          Nivell confiança cants curts amb cant llarg previ    -1    0,4    0,1    -
          Freqüència mitjana                        -0,85    11750    13250    17500    19000
          Nivell confiança grups de polsos per segon
            -0,05    1    0    -
          Math.min(2, batecs ales / tempsDurada)
            -0,1    2    0    -
          Proporció silencis                        -0,8    0,5    0,75    1    -

          Mínima frequenciaMaximaMostreig = 11000
          Confiança freqüència = (1.5-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),2.7))/1.5
          freqüència += 8000*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),2.7)
        */

        double semblancaCicadettaCerdaniensis = 1.0;

        double importanciaBaseMitjanaDuradaCantCurt = 0.75;
        double importanciaBaseSonCantsCurts = 1.0;
        double importanciaBaseMitjanaDuradaSilencis = 0.6;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
        double importanciaBaseMitjanaFrequencia = 0.85;
        double importanciaBaseNivellConfiancaNGP = 0.05;
        double importanciaBaseBatecsAles = 0.1;
        double importanciaBaseProporcioSilenci = 0.8;

        frequenciaMitjana += 8000*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),2.7);

        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 11000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;


        double valorMitjanaDuradaCantCurt = Math.pow(((duradaMitjanaCantCurt < 30) ? Math.max(0, 1.0*(duradaMitjanaCantCurt - 20) / (30-20)) : ((duradaMitjanaCantCurt > 180) ? Math.max(0, 1 - 1.0*(duradaMitjanaCantCurt - 180) / (300-180)) : 1)), potencia);
        double valorSonCantsCurts = Math.pow(Math.max(0,Math.min(1, (nivellConfiancaSonCantsCurts - 0.75) / (1-0.75))), potencia);
        double valorMitjanaDuradaSilencis = Math.pow(((duradaMitjanaSilenci < 170) ? Math.max(0, 1.0*(duradaMitjanaSilenci - 100) / (170-100)) : ((duradaMitjanaSilenci > 1500) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenci - 1500) / (2000-1500)) : 1)), potencia);
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 13250) ? Math.max(0, 1.0*(frequenciaMitjana - 11750) / (13250-11750)) : ((frequenciaMitjana > 17500) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 17500) / (19000-17500)) : 1)), potencia);
        double valorNivellConfiancaNGP = 1 - nivellConfiancaNombreGrupsDePolsosPerSegon;
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(((proporcioSilenci < 0.75) ? Math.max(0, 1.0*(proporcioSilenci - 0.5) / (0.75-0.5)) : 1), potencia);

        semblancaCicadettaCerdaniensis -= (1 - valorMitjanaDuradaCantCurt) * importanciaBaseMitjanaDuradaCantCurt;
        semblancaCicadettaCerdaniensis -= (1 - valorSonCantsCurts) * importanciaBaseSonCantsCurts;
        semblancaCicadettaCerdaniensis -= (1 - valorMitjanaDuradaSilencis) * importanciaBaseMitjanaDuradaSilencis;
        semblancaCicadettaCerdaniensis -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        if (fiabilitatIdentificacio != -1) semblancaCicadettaCerdaniensis -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        semblancaCicadettaCerdaniensis -= (1 - valorNivellConfiancaNGP) * importanciaBaseNivellConfiancaNGP;
        semblancaCicadettaCerdaniensis -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaCicadettaCerdaniensis -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaCicadettaCerdaniensis = Math.max(0,semblancaCicadettaCerdaniensis);
        return new Pair(semblancaCicadettaCerdaniensis, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Hilaphura varipes") == 0) {
        /*
        //Hilaphura varipes
        Criteris:
                                    Importància base    0%    100%    100%    0%
          Durada mitjana cant curt                  -0,9    120    170    360    450
          Nivell confiança són cants curts          -1    0    1    -
          Durada mitjana silenci                    -0,75    20    30    220    280
          Nivell confiança cants curts amb cant llarg previ    -1    0,4    0,1    0    -
          Pendent primera meitat cant curt          -0,4    0,9    1,2    1,7    2,2
          Pendent segona meitat cant curt           -0,4    0,7    0,95    1,15    1,5
          Pendent primera meitat cant curt - pendent segona meitat cant curt    -0,3    0    0,15    1    -
          Correlació potència cants i llargada silencis    -0,25    -0,25    0,25    1    -
          Freqüència mitjana                        -1    4750    6750    8750    10750
          Nivell confiança grups de polsos per segon
            -0,6    1    0    -
          Math.min(2, batecs ales / tempsDurada)
            -0,2    2    0    -
          Proporció silencis                        -0,5    0    0,2    0,45    0.7

          Mínima frequenciaMaximaMostreig = 11000
          Confiança freqüència = (4-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),5))/4
          freqüència += 2200*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),5)
        */

        double semblancaHilaphuraVaripes = 1.0;

        double importanciaBaseMitjanaDuradaCantCurt = 0.9;
        double importanciaBaseSonCantsCurts = 1.0;
        double importanciaBaseMitjanaDuradaSilencis = 0.75;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
        double importanciaBasePendentPrimeraMeitatCantCurt = 0.4;
        double importanciaBasePendentSegonaMeitatCantCurt = 0.4;
        double importanciaBasePendentSegonaMeitatMenysPendentPrimeraMeitatCantCurt = 0.3;
        double importanciaBaseCorrelacioPotenciaCantsILlargadaSilencis = 0.25;
        double importanciaBaseMitjanaFrequencia = 1.0;
        double importanciaBaseNivellConfiancaNGP = 0.6;
        double importanciaBaseBatecsAles = 0.2;
        double importanciaBaseProporcioSilenci = 0.5;

        frequenciaMitjana += 2200*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),5);
```

```
        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 11000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;


        double valorMitjanaDuradaCantCurt = Math.pow(((duradaMitjanaCantCurt < 170) ? Math.max(0, 1.0*(duradaMitjanaCantCurt - 120) / (170-120)) : ((duradaMitjanaCantCurt > 360) ? Math.max(0, 1 - 1.0*(duradaMitjanaCantCurt - 360) / (450-360)) : 1)), potencia);
        double valorSonCantsCurts = Math.pow(Math.min(1, nivellConfiancaSonCantsCurts), potencia);
        double valorMitjanaDuradaSilencis = Math.pow(((duradaMitjanaSilenci < 30) ? Math.max(0, 1.0*(duradaMitjanaSilenci - 20) / (30-20)) : ((duradaMitjanaSilenci > 220) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenci - 220) / (280-220)) : 1)), potencia);
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorPendentPrimeraMeitatCantCurt = Math.pow(((pendentPrimeraMeitatCantsCurts < 1.2) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsCurts - 0.9) / (1.2-0.9)) : ((pendentPrimeraMeitatCantsCurts > 1.7) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsCurts - 1.7) / (2.2-1.7)) : 1)), potencia);
        double valorPendentSegonaMeitatCantCurt = Math.pow(((pendentSegonaMeitatCantsCurts < 0.95) ? Math.max(0, 1.0*(pendentSegonaMeitatCantsCurts - 0.7) / (0.95-0.7)) : ((pendentSegonaMeitatCantsCurts > 1.15) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsCurts - 1.15) / (1.5-1.15)) : 1)), potencia);
        double valorPendentSegonaMeitatMenysPendentPrimeraMeitatCantCurt = Math.pow(Math.min(1, Math.max(0, (pendentPrimeraMeitatCantsCurts - pendentSegonaMeitatCantsCurts) / (0.15))), potencia);
        double valorCorrelacioPotenciaCantsILlargadaSilencis = Math.pow(Math.min(1, Math.max(0, (correlacioPotenciaCantsILlargadaSilencis + 0.25) / (0.5))), potencia);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 6750) ? Math.max(0, 1.0*(frequenciaMitjana - 4750) / (6750-4750)) : ((frequenciaMitjana > 8750) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 8750) / (10750-8750)) : 1)), potencia);
        double valorNivellConfiancaNGP = 1 - nivellConfiancaNombreGrupsDePolsosPerSegon;
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(((proporcioSilenci < 0.2) ? Math.max(0, 1.0*(proporcioSilenci - 0.0) / (0.2-0.0)) : ((proporcioSilenci > 0.45) ? Math.max(0, 1 - 1.0*(proporcioSilenci - 0.45) / (0.7-0.45)) : 1)), potencia);

        semblancaHilaphuraVaripes -= (1 - valorMitjanaDuradaCantCurt) * importanciaBaseMitjanaDuradaCantCurt;
        semblancaHilaphuraVaripes -= (1 - valorSonCantsCurts) * importanciaBaseSonCantsCurts;
        semblancaHilaphuraVaripes -= (1 - valorMitjanaDuradaSilencis) * importanciaBaseMitjanaDuradaSilencis;
        semblancaHilaphuraVaripes -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        semblancaHilaphuraVaripes -= (1 - valorPendentPrimeraMeitatCantCurt) * importanciaBasePendentPrimeraMeitatCantCurt;
        semblancaHilaphuraVaripes -= (1 - valorPendentSegonaMeitatCantCurt) * importanciaBasePendentSegonaMeitatCantCurt;
        semblancaHilaphuraVaripes -= (1 - valorPendentSegonaMeitatMenysPendentPrimeraMeitatCantCurt) * importanciaBasePendentSegonaMeitatMenysPendentPrimeraMeitatCantCurt;
        semblancaHilaphuraVaripes -= (1 - valorCorrelacioPotenciaCantsILlargadaSilencis) * importanciaBaseCorrelacioPotenciaCantsILlargadaSilencis;
        if (fiabilitatIdentificacio != -1) semblancaHilaphuraVaripes -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        semblancaHilaphuraVaripes -= (1 - valorNivellConfiancaNGP) * importanciaBaseNivellConfiancaNGP;
        semblancaHilaphuraVaripes -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaHilaphuraVaripes -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaHilaphuraVaripes = Math.max(0,semblancaHilaphuraVaripes);
        return new Pair(semblancaHilaphuraVaripes, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Lyristes plebejus") == 0) {

        /*
        //Lyristes plebejus
        Criteris:
                                    Importància base    0%      100%    100%    0%
          Mitjana durada cant curt                  -1      20      35      80      100
          Mitjana durada silenci                    -0,6    0       20      60      100
          Nivell confiança cants curts amb cant llarg previ -0,8    0,4     0,1     0       -
          Pendent primera meitat cant llarg         -0,2    0,65    0,9     1,1     1,5
          Pendent segona meitat cant llarg          -0,2    0,65    0,9     1,1     1,5
          Freqüència mitjana                        -1,0    5000    6500    8750    10250
          Percentatge de cants curts                -0,5    "=100"  "<100"  -
          Percentatge de cants curts 2              -0,25   0,7     0,85    1       -
          Nombre grups de polsos != corsica fairmairei -0,15 59 <= NGP <= 65  NGP < 59 || NGP > 65  -
          Nombre grups de polsos != garricola       -0,125  66 <= NGP <= 71  NGP < 66 || NGP > 71  -
          Nombre grups de polsos != haematodes      -0,075  90 <= NGP <= 105  NGP < 90 || NGP > 105  -
          Nombre grups de polsos != quadrisignata   -0,1    73 <= NGP <= 80  NGP < 73 || NGP > 80  -
          Math.min(2, batecs ales / tempsDurada)    -0,25   2       0       -
          Proporció silencis                        -0,2    0,65    0,35    0       -

          Mínima frequenciaMaximaMostreig = 7000
          Confiança freqüència = (4-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-7000)/17000),5))/4
          freqüència += 3000*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-7000)/17000),5)
        */

        double semblancaLyristesPlebejus = 1.0;

        double importanciaBaseMitjanaDuradaCantsCurts = 1.0;
        double importanciaBaseMitjanaDuradaSilencis = 0.6;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 0.8;
        double importanciaBasePendentPrimeraMeitatCantLlarg = 0.2;
        double importanciaBasePendentSegonaMeitatCantLlarg = 0.2;
        double importanciaBaseMitjanaFrequencia = 1.0;
        double importanciaBasePercentatgeCantsCurts = 0.5;
        double importanciaBasePercentatgeCantsCurts2 = 0.25;
        double importanciaBaseNGPDiferentCorsicaFairmairei = 0.15;
        double importanciaBaseNGPDiferentGarricola = 0.125;
        double importanciaBaseNGPDiferentHaematodes = 0.075;
        double importanciaBaseNGPDiferentQuadrisignata = 0.1;
        double importanciaBaseBatecsAles = 0.25;
        double importanciaBaseProporcioSilenci = 0.2;


        frequenciaMitjana += 3000*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-7000)/17000),5);

        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 7000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;


        boolean percentatgeCantsCurtsCorrecte = percentatgeCantsCurts > 85 || nCants < 2;


        double valorMitjanaDuradaCantsCurts = Math.pow(((duradaMitjanaCantCurt < 35) ? Math.max(0, 1.0*(duradaMitjanaCantCurt - 20) / (35-20)) : ((duradaMitjanaCantCurt > 80) ? Math.max(0, 1 - 1.0*(duradaMitjanaCantCurt - 80) / (100-80)) : 1)), potencia);
        double valorMitjanaDuradaSilencis = Math.pow(((duradaMitjanaSilenci < 20) ? Math.max(0, 1.0*(duradaMitjanaSilenci - 0) / (20-0)) : ((duradaMitjanaSilenci > 60) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenci - 60) / (100-60)) : 1)), potencia);
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorPendentPrimeraMeitatCantLlarg = Math.pow(((pendentPrimeraMeitatCantsLlargs < 0.9) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsLlargs - 0.65) / (0.9-0.65)) : ((pendentPrimeraMeitatCantsLlargs > 1.1) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsLlargs - 1.1) / (1.5-1.1)) : 1)), potencia);
        double valorPendentSegonaMeitatCantLlarg = Math.pow(((pendentSegonaMeitatCantsLlargs < 0.9) ? Math.max(0, 1.0*(pendentSegonaMeitatCantsLlargs - 0.65) / (0.9-0.65)) : ((pendentSegonaMeitatCantsLlargs > 1.1) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsLlargs - 1.1) / (1.5-1.1)) : 1)), potencia);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 6500) ? Math.max(0, 1.0*(frequenciaMitjana - 5000) / (6500-5000)) : ((frequenciaMitjana > 8750) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 8750) / (10250-8750)) : 1)), potencia);
        double valorPercentatgeCantsCurts = percentatgeCantsCurts != 100 ? 1 : 0;
        double valorPercentatgeCantsCurts2 = percentatgeCantsCurtsCorrecte ? (Math.pow(Math.min(1, Math.max(0, (percentatgeCantsCurts - 70) / 15)), potencia)) : 0;
```

```
        double valorNGPDiferentCorsicaFairmairei = (nombreGrupsDePolsosPerSegon >= 59 && nombreGrupsDePolsosPerSegon <= 65) ? 0 : 1;
        double valorNGPDiferentGarricola = (nombreGrupsDePolsosPerSegon >= 66 && nombreGrupsDePolsosPerSegon <= 71) ? 0 : 1;
        double valorNGPDiferentHaematodes = (nombreGrupsDePolsosPerSegon >= 90 && nombreGrupsDePolsosPerSegon <= 105) ? 0 : 1;
        double valorNGPDiferentQuadrisignata = (nombreGrupsDePolsosPerSegon >= 73 && nombreGrupsDePolsosPerSegon <= 80) ? 0 : 1;
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(Math.min(1, Math.max(0, (1 - proporcioSilenci - 0.35) / (0.3))), potencia);



        if (percentatgeCantsCurts > 0) semblancaLyristesPlebejus -= (1 - valorMitjanaDuradaCantsCurts) * importanciaBaseMitjanaDuradaCantsCurts;
        if (percentatgeCantsCurts > 0) semblancaLyristesPlebejus -= (1 - valorMitjanaDuradaSilencis) * importanciaBaseMitjanaDuradaSilencis;
        semblancaLyristesPlebejus -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        if (percentatgeCantsCurts < 100) semblancaLyristesPlebejus -= (1 - valorPendentPrimeraMeitatCantLlarg) * importanciaBasePendentPrimeraMeitatCantLlarg;
        if (percentatgeCantsCurts < 100) semblancaLyristesPlebejus -= (1 - valorPendentSegonaMeitatCantLlarg) * importanciaBasePendentSegonaMeitatCantLlarg;
        if (fiabilitatIdentificacio != -1) semblancaLyristesPlebejus -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        semblancaLyristesPlebejus -= (1 - valorPercentatgeCantsCurts) * importanciaBasePercentatgeCantsCurts;
        semblancaLyristesPlebejus -= (1 - valorPercentatgeCantsCurts2) * importanciaBasePercentatgeCantsCurts2;
        semblancaLyristesPlebejus -= (1 - valorNGPDiferentCorsicaFairmairei) * importanciaBaseNGPDiferentCorsicaFairmairei;
        semblancaLyristesPlebejus -= (1 - valorNGPDiferentGarricola) * importanciaBaseNGPDiferentGarricola;
        semblancaLyristesPlebejus -= (1 - valorNGPDiferentHaematodes) * importanciaBaseNGPDiferentHaematodes;
        semblancaLyristesPlebejus -= (1 - valorNGPDiferentQuadrisignata) * importanciaBaseNGPDiferentQuadrisignata;
        semblancaLyristesPlebejus -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaLyristesPlebejus -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaLyristesPlebejus = Math.max(0,semblancaLyristesPlebejus);
        return new Pair(semblancaLyristesPlebejus, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Tettigettalna argentata") == 0) {
        /*
        //Tettigettalna argentata
        Criteris:
                                        Importància base    0%      100%    100%    0%
        Mitjana durada cant curt                -1          10      15      40      70
        Nivell confiança són cants curts        -1          0,75    1       -
        Mitjana durada silenci                  -0,3        10      55      85      140
        Nivell confiança cants curts amb cant llarg previ       -1      0,4     0,1     0       -
        Freqüència mitjana                      -0,5        7500    9500    12500   14500
        Nivell confiança grups de polsos per segon      -0,6    1       0       -
        Math.min(2, batecs ales / tempsDurada)          -1      2       0       -
        Proporció silencis                      -0,65       0,5     0,7     0,9     1

        Mínima frequenciaMaximaMostreig = 11000
        Confiança freqüència = (3-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),4))/3
        freqüència += 3700*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),4)
        */

        double semblancaTettigettalnaArgentata = 1.0;

        double importanciaBaseMitjanaDuradaCantCurt = 1.0;
        double importanciaBaseSonCantsCurts = 1.0;
        double importanciaBaseMitjanaDuradaSilencis = 0.3;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
        double importanciaBaseMitjanaFrequencia = 0.5;
        double importanciaBaseNivellConfiancaNGP = 0.6;
        double importanciaBaseBatecsAles = 1.0;
        double importanciaBaseProporcioSilenci = 0.65;

        frequenciaMitjana += 3700*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),4);

        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 11000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;



        double valorMitjanaDuradaCantCurt = Math.pow(((duradaMitjanaCantCurt < 15) ? Math.max(0, 1.0*(duradaMitjanaCantCurt - 10) / (15-10)) : ((duradaMitjanaCantCurt > 40) ? Math.max(0, 1 - 1.0*(duradaMitjanaCantCurt - 40) / (70-40)) : 1)), potencia);
        double valorSonCantsCurts = Math.pow(Math.max(0,Math.min(1, (nivellConfiancaSonCantsCurts - 0.75) / (1-0.75))), potencia);
        double valorMitjanaDuradaSilencis = Math.pow(((duradaMitjanaSilenci < 55) ? Math.max(0, 1.0*(duradaMitjanaSilenci - 10) / (55-10)) : ((duradaMitjanaSilenci > 85) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenci - 85) / (140-85)) : 1)), potencia);
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 9500) ? Math.max(0, 1.0*(frequenciaMitjana - 7500) / (9500-7500)) : ((frequenciaMitjana > 12500) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 12500) / (14500-12500)) : 1)), potencia);
        double valorNivellConfiancaNGP = 1 - nivellConfiancaNombreGrupsDePolsosPerSegon;
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(((proporcioSilenci < 0.7) ? Math.max(0, 1.0*(proporcioSilenci - 0.5) / (0.7-0.5)) : ((proporcioSilenci > 0.9) ? Math.max(0, 1 - 1.0*(proporcioSilenci - 0.9) / (1-0.9)) : 1)), potencia);

        semblancaTettigettalnaArgentata -= (1 - valorMitjanaDuradaCantCurt) * importanciaBaseMitjanaDuradaCantCurt;
        semblancaTettigettalnaArgentata -= (1 - valorSonCantsCurts) * importanciaBaseSonCantsCurts;
        semblancaTettigettalnaArgentata -= (1 - valorMitjanaDuradaSilencis) * importanciaBaseMitjanaDuradaSilencis;
        semblancaTettigettalnaArgentata -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        if (fiabilitatIdentificacio != -1) semblancaTettigettalnaArgentata -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        semblancaTettigettalnaArgentata -= (1 - valorNivellConfiancaNGP) * importanciaBaseNivellConfiancaNGP;
        semblancaTettigettalnaArgentata -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaTettigettalnaArgentata -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaTettigettalnaArgentata = Math.max(0,semblancaTettigettalnaArgentata);
        return new Pair(semblancaTettigettalnaArgentata, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Tettigettula pygmea") == 0) {
        /*
        //Tettigettula pygmea
        Criteris:
                                        Importància base    0%      100%    100%    0%
        Mitjana durada cant curt                -0,75       110     170     260     450
        Nivell confiança són cants curts        -1          0       1       -
        Mitjana durada silenci                  -0,2        40      80      160     280
        Durada mitjana cant curt amb cant llarg previ       -1      30      45      80      95
        Durada mitjana cant llarg amb cant curt posterior   -1      300     370     530     600
        Durada silenci entre cant curt i cant llarg     -1      10      20      50      70
        Freqüència mitjana                      -1          14500   16500   18500   20500
        Math.min(2, batecs ales / tempsDurada)      -0,1    0       0,1     -
        Nombre grups de polsos != corsica fairmairei    -0,15   59 <= NGP <= 65     NGP < 59 || NGP > 65    -
        Nombre grups de polsos != garricola     -0,125  66 <= NGP <= 71      NGP < 66 || NGP > 71    -
        Nombre grups de polsos != haematodes        -0,075  90 <= NGP <= 105    NGP < 90 || NGP > 105   -
        Nombre grups de polsos != quadrisignata     -0,1    73 <= NGP <= 80NGP < 73 || NGP > 80     -
        Proporció silencis                      -0,25       0       0,25    0,65    1

        Mínima frequenciaMaximaMostreig = 16000
        Confiança freqüència = (3-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-16000)/8000),4))/3
```

```
            freqüència += 4500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-16000)/8000),4)
            */

            double semblancaTettigettulaPygmea = 1.0;

            double importanciaBaseMitjanaDuradaCantCurt = 0.75;
            double importanciaBaseSonCantsCurts = 1.0;
            double importanciaBaseMitjanaDuradaSilencis = 0.2;
            double importanciaBaseDuradaCantCurtAmbCantLlargPrevi = 1.0;
            double importanciaBaseDuradaCantLlargAmbCantCurtPosterior = 1.0;
            double importanciaBaseDuradaSilenciEntreCantLlargICantCurtPosterior = 1.0;
            double importanciaBaseMitjanaFrequencia = 1.0;
            double importanciaBaseBatecsAles = 0.1;
            double importanciaBaseNGPDiferentCorsicaFairmairei = 0.15;
            double importanciaBaseNGPDiferentGarricola = 0.125;
            double importanciaBaseNGPDiferentHaematodes = 0.075;
            double importanciaBaseNGPDiferentQuadrisignata = 0.1;
            double importanciaBaseProporcioSilenci = 0.25;

            frequenciaMitjana += 4500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-16000)/8000),4);

            double fiabilitatIdentificacio;
            if (frequenciaMaximaMostreig < 16000) fiabilitatIdentificacio = -1.0;
            else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
            else fiabilitatIdentificacio = 1.0;


            double valorMitjanaDuradaCantCurt = Math.pow(((duradaMitjanaCantCurt < 170) ? Math.max(0, 1.0*(duradaMitjanaCantCurt - 110) / (170-110)) : ((duradaMitjanaCantCurt > 260) ? Math.max(0, 1 - 1.0*(duradaMitjanaCantCurt - 260) / (450-260)) : 1)), potencia);
            double valorSonCantsCurts = Math.pow(Math.min(1, nivellConfiancaSonCantsCurts), potencia);
            double valorMitjanaDuradaSilencis = Math.pow(((duradaMitjanaSilenci < 80) ? Math.max(0, 1.0*(duradaMitjanaSilenci - 40) / (80-40)) : ((duradaMitjanaSilenci > 160) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenci - 160) / (280-160)) : 1)), potencia);
            double valorDuradaCantCurtAmbCantLlargPrevi = Math.pow(((duradaMitjanaCantCurtAmbCantLlargPrevi < 45) ? Math.max(0, 1.0*(duradaMitjanaCantCurtAmbCantLlargPrevi - 30) / (45-30)) : ((duradaMitjanaCantCurtAmbCantLlargPrevi > 80) ? Math.max(0, 1 - 1.0*(duradaMitjanaCantCurtAmbCantLlargPrevi - 80) / (95-80)) : 1)), potencia);
            double valorDuradaCantLlargAmbCantCurtPosterior = Math.pow(((duradaMitjanaCantLlargAmbCantCurtPosterior < 370) ? Math.max(0, 1.0*(duradaMitjanaCantLlargAmbCantCurtPosterior - 300) / (370-300)) : ((duradaMitjanaCantLlargAmbCantCurtPosterior > 530) ? Math.max(0, 1 - 1.0*(duradaMitjanaCantLlargAmbCantCurtPosterior - 530) / (600-530)) : 1)), potencia);
            double valorDuradaSilenciEntreCantLlargICantCurtPosterior = Math.pow(((duradaMitjanaSilenciEntreCantLlargICantCurtPosterior < 20) ? Math.max(0, 1.0*(duradaMitjanaSilenciEntreCantLlargICantCurtPosterior - 10) / (20-10)) : ((duradaMitjanaSilenciEntreCantLlargICantCurtPosterior > 50) ? Math.max(0, 1 - 1.0*(duradaMitjanaSilenciEntreCantLlargICantCurtPosterior - 50) / (70-50)) : 1)), potencia);
            double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 16500) ? Math.max(0, 1.0*(frequenciaMitjana - 14500) / (16500-14500)) : ((frequenciaMitjana > 18500) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 18500) / (20500-18500)) : 1)), potencia);
            double valorBatecsAles = Math.min(1, (nBatecsAlesOCantsIncorrectes / tempsDurada) / 0.1);
            double valorNGPDiferentCorsicaFairmairei = (nombreGrupsDePolsosPerSegon >= 59 && nombreGrupsDePolsosPerSegon <= 65) ? 0 : 1;
            double valorNGPDiferentGarricola = (nombreGrupsDePolsosPerSegon >= 66 && nombreGrupsDePolsosPerSegon <= 71) ? 0 : 1;
            double valorNGPDiferentHaematodes = (nombreGrupsDePolsosPerSegon >= 90 && nombreGrupsDePolsosPerSegon <= 105) ? 0 : 1;
            double valorNGPDiferentQuadrisignata = (nombreGrupsDePolsosPerSegon >= 73 && nombreGrupsDePolsosPerSegon <= 80) ? 0 : 1;
            double valorProporcioSilenci = Math.pow(((proporcioSilenci < 0.25) ? Math.max(0, 1.0*(proporcioSilenci - 0) / (0.25-0)) : ((proporcioSilenci > 0.65) ? Math.max(0, 1 - 1.0*(proporcioSilenci - 0.65) / (1-0.65)) : 1)), potencia);

            semblancaTettigettulaPygmea -= (1 - valorMitjanaDuradaCantCurt) * importanciaBaseMitjanaDuradaCantCurt;
            semblancaTettigettulaPygmea -= (1 - valorSonCantsCurts) * importanciaBaseSonCantsCurts;
            semblancaTettigettulaPygmea -= (1 - valorMitjanaDuradaSilencis) * importanciaBaseMitjanaDuradaSilencis;
            if (nivellConfiancaCantsCurtsAmbCantLlargPrevi > 0.2) semblancaTettigettulaPygmea -= (1 - valorDuradaCantCurtAmbCantLlargPrevi) * importanciaBaseDuradaCantCurtAmbCantLlargPrevi;
            if (nivellConfiancaCantsCurtsAmbCantLlargPrevi > 0.2) semblancaTettigettulaPygmea -= (1 - valorDuradaCantLlargAmbCantCurtPosterior) * importanciaBaseDuradaCantLlargAmbCantCurtPosterior;
            if (nivellConfiancaCantsCurtsAmbCantLlargPrevi > 0.2) semblancaTettigettulaPygmea -= (1 - valorDuradaSilenciEntreCantLlargICantCurtPosterior) * importanciaBaseDuradaSilenciEntreCantLlargICantCurtPosterior;
            if (fiabilitatIdentificacio != -1) semblancaTettigettulaPygmea -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
            semblancaTettigettulaPygmea -= (1 - valorBatecsAles) * importanciaBaseBatecsAles;
            semblancaTettigettulaPygmea -= (1 - valorNGPDiferentCorsicaFairmairei) * importanciaBaseNGPDiferentCorsicaFairmairei;
            semblancaTettigettulaPygmea -= (1 - valorNGPDiferentGarricola) * importanciaBaseNGPDiferentGarricola;
            semblancaTettigettulaPygmea -= (1 - valorNGPDiferentHaematodes) * importanciaBaseNGPDiferentHaematodes;
            semblancaTettigettulaPygmea -= (1 - valorNGPDiferentQuadrisignata) * importanciaBaseNGPDiferentQuadrisignata;
            semblancaTettigettulaPygmea -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

            semblancaTettigettulaPygmea = Math.max(0,semblancaTettigettulaPygmea);
            return new Pair(semblancaTettigettulaPygmea, fiabilitatIdentificacio);
        }
        else if (especie.compareTo("Tibicina corsica fairmairei") == 0) {
            /*
            //Tibicina corsica fairmairei
            Criteris:
                                                Importància base    0%      100%    100%    0%
            Nivell confiança cants curts amb cant llarg previ       -1      0,4     0,1     0       -
            Pendent primera meitat cant llarg       -0,2    0,85    1       1,2     1,4
            Pendent segona meitat cant llarg        -0,3    0,85    1       2       2,5
            Nivell confiança cant llarg irregular   -0,1    0,5     0
            Freqüència mitjana      -0,9    7500    9000    10500   12000
            Pic 1 freqüència        -0,15   7250    8250    9750    10750
            Pic 2 freqüència        -0,15   7750    8750    11000   12000
            Pic 3 freqüència        -0,15   8750    9750    11500   12500
            Nombre de grups de polsos per segon     -1      56      59      65      68
            Definició grups de polsos per segon     -0,1    0,3     0,2     0       -
            Math.min(2, batecs ales / tempsDurada)
                -0,2    2       0       -
            Proporció silencis      -0,1    1       0,5     0       -

            Mínima frequenciaMaximaMostreig = 9000
            Mínima frequenciaMaximaMostreig pels 3 pics = 12000
            Confiança freqüència = (2-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),10))/2
            freqüència += 1500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),10)
            */

            double semblancaTibicinaCorsicaFairmairei = 1.0;

            double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
            double importanciaBasePendentPrimeraMeitatCantLlarg = 0.2;
            double importanciaBasePendentSegonaMeitatCantLlarg = 0.3;
            double importanciaBaseCantLlargIrregular = 0.1;
            double importanciaBaseMitjanaFrequencia = 0.9;
            double importanciaBasePic1Frequencia = 0.15;
            double importanciaBasePic2Frequencia = 0.15;
            double importanciaBasePic3Frequencia = 0.15;
            double importanciaBaseNGP = 1.0;
            double importanciaBaseDefinicioNGP = 0.1;
            double importanciaBaseBatecsAles = 0.2;
            double importanciaBaseProporcioSilenci = 0.1;

            frequenciaMitjana += 1500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),10);

            double fiabilitatIdentificacio;
            if (frequenciaMaximaMostreig < 9000) fiabilitatIdentificacio = -1.0;
            else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
            else fiabilitatIdentificacio = 1.0;


            double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
            double valorPendentPrimeraMeitatCantLlarg = Math.pow(((pendentPrimeraMeitatCantsLlargs < 1.0) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsLlargs - 0.85) / (1.0-0.85)) : ((pendentPrimeraMeitatCantsLlargs > 1.2) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsLlargs - 1.2) / (1.4-1.2)) : 1)), potencia);
            double valorPendentSegonaMeitatCantLlarg = Math.pow(((pendentSegonaMeitatCantsLlargs < 1.0) ? Math.max(0,
```

```
1.0*(pendentSegonaMeitatCantsLlargs - 0.85) / (1.0-0.85)) : ((pendentSegonaMeitatCantsLlargs > 2.0) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsLlargs - 2.0) / (2.5-2.0)) : 1)), potencia);
        double valorEsCantLlargIrregular = Math.min(1, 1 - nivellConfiancaEsCantLlargIrregular + 0.5);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 9000) ? Math.max(0, 1.0*(frequenciaMitjana - 7500) / (9000-7500)) : ((frequenciaMitjana > 10500) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 10500) / (12000-10500)) : 1)), potencia);
        double valorPic1Frequencia = Math.pow(((pic1Frequencia < 8250) ? Math.max(0, 1.0*(pic1Frequencia - 7250) / (8250-7250)) : ((pic1Frequencia > 9750) ? Math.max(0, 1 - 1.0*(pic1Frequencia - 9750) / (10750-9750)) : 1)), potencia);
        double valorPic2Frequencia = Math.pow(((pic2Frequencia < 8750) ? Math.max(0, 1.0*(pic2Frequencia - 7750) / (8750-7750)) : ((pic2Frequencia > 11000) ? Math.max(0, 1 - 1.0*(pic2Frequencia - 11000) / (12000-11000)) : 1)), potencia);
        double valorPic3Frequencia = Math.pow(((pic3Frequencia < 9750) ? Math.max(0, 1.0*(pic3Frequencia - 8750) / (9750-8750)) : ((pic3Frequencia > 11500) ? Math.max(0, 1 - 1.0*(pic3Frequencia - 11500) / (12500-11500)) : 1)), potencia);
        double valorNGP = Math.pow(((nombreGrupsDePolsosPerSegon < 59) ? Math.max(0, 1.0*(nombreGrupsDePolsosPerSegon - 56) / (59-56)) : ((nombreGrupsDePolsosPerSegon > 65) ? Math.max(0, 1 - 1.0*(nombreGrupsDePolsosPerSegon - 65) / (68-65)) : 1)), potencia);
        double valorDefinicioNGP = Math.pow(Math.min(1, Math.max(0, (1 - definicioGrupsDePolsos - 0.7) / (0.1))), potencia);
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(Math.min(1, Math.max(0, (1 - proporcioSilenci - 0.5) / (0.5))), potencia);


        semblancaTibicinaCorsicaFairmairei -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        semblancaTibicinaCorsicaFairmairei -= (1 - valorPendentPrimeraMeitatCantLlarg) * importanciaBasePendentPrimeraMeitatCantLlarg;
        semblancaTibicinaCorsicaFairmairei -= (1 - valorPendentSegonaMeitatCantLlarg) * importanciaBasePendentSegonaMeitatCantLlarg;
        semblancaTibicinaCorsicaFairmairei -= (1 - valorEsCantLlargIrregular) * importanciaBaseCantLlargIrregular;
        if (fiabilitatIdentificacio != -1) semblancaTibicinaCorsicaFairmairei -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        if (frequenciaMaximaMostreig >= 12000) semblancaTibicinaCorsicaFairmairei -= (1 - valorPic1Frequencia) * importanciaBasePic1Frequencia;
        if (frequenciaMaximaMostreig >= 12000) semblancaTibicinaCorsicaFairmairei -= (1 - valorPic2Frequencia) * importanciaBasePic2Frequencia;
        if (frequenciaMaximaMostreig >= 12000) semblancaTibicinaCorsicaFairmairei -= (1 - valorPic3Frequencia) * importanciaBasePic3Frequencia;
        semblancaTibicinaCorsicaFairmairei -= (1 - valorNGP) * importanciaBaseNGP;
        semblancaTibicinaCorsicaFairmairei -= (1 - valorDefinicioNGP) * importanciaBaseDefinicioNGP;
        semblancaTibicinaCorsicaFairmairei -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaTibicinaCorsicaFairmairei -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaTibicinaCorsicaFairmairei = Math.max(0,semblancaTibicinaCorsicaFairmairei);
        return new Pair(semblancaTibicinaCorsicaFairmairei, fiabilitatIdentificacio);
    }
    else if (especie.compareTo("Tibicina garricola") == 0) {
      /*
      //Tibicina garricola
      Criteris:
                                    Importància base    0%      100%    100%    0%
        Nivell confiança cants curts amb cant llarg previ       -1      0,4     0,1     0       -
        Nivell confiança són cants llargs       -0,75   0       1       -
        Pendent primera meitat cant llarg       -0,2    0,8     0,95    1,3     1,6
        Pendent segona meitat cant llarg        -0,3    0,8     0,95    1,2     1,4
        Nivell confiança cant llarg irregular   -0,3    0,5     0       -
        Freqüència mitjana      -0,8    6250    7750    9250    10750
        Pic 1 freqüència        -0,15   6000    7000    8750    9750
        Pic 2 freqüència        -0,15   6750    7750    9500    10500
        Pic 3 freqüència        -0,15   7750    8750    10250   11250
        Nombre de grups de polsos per segon     -1      63      66      71      74
        Definició grups de polsos per segon     -0,1    0,8     0,35    0       -
        Math.max(2, batecs ales / tempsDurada)
          -0,2    2       0       -
        Proporció silencis      -0,1    1       0,4     0       -
            Mínima frequenciaMaximaMostreig = 9000
            Mínima frequenciaMaximaMostreig pels 3 pics = 11000
            Confiança freqüència = (2-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),6))/2
            freqüència += 1200*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),6)
        */

        double semblancaTibicinaGarricola = 1.0;

        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
        double importanciaBaseSonCantsLlargs = 0.75;
        double importanciaBasePendentPrimeraMeitatCantLlarg = 0.2;
        double importanciaBasePendentSegonaMeitatCantLlarg = 0.3;
        double importanciaBaseCantLlargIrregular = 0.3;
        double importanciaBaseMitjanaFrequencia = 0.8;
        double importanciaBasePic1Frequencia = 0.15;
        double importanciaBasePic2Frequencia = 0.15;
        double importanciaBasePic3Frequencia = 0.15;
        double importanciaBaseNGP = 1.0;
        double importanciaBaseDefinicioNGP = 0.1;
        double importanciaBaseBatecsAles = 0.2;
        double importanciaBaseProporcioSilenci = 0.1;


        frequenciaMitjana += 1200*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),6);

        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 9000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;

        //No és habitual, però pot ser que, en alguns cants, em divideixi els polsos en dos.
        double ngpAux = nombreGrupsDePolsosPerSegon;
        if (nombreGrupsDePolsosPerSegon > 132 && nombreGrupsDePolsosPerSegon < 142) {
          nombreGrupsDePolsosPerSegon /= 2;
        }

        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorSonCantsLlargs = Math.pow(nivellConfiancaSonCantsLlargs, potencia);
        double valorPendentPrimeraMeitatCantLlarg = Math.pow(((pendentPrimeraMeitatCantsLlargs < 0.95) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsLlargs - 0.8) / (0.95-0.8)) : ((pendentPrimeraMeitatCantsLlargs > 1.3) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsLlargs - 1.3) / (1.6-1.3)) : 1)), potencia);
        double valorPendentSegonaMeitatCantLlarg = Math.pow(((pendentSegonaMeitatCantsLlargs < 0.95) ? Math.max(0, 1.0*(pendentSegonaMeitatCantsLlargs - 0.8) / (0.95-0.8)) : ((pendentSegonaMeitatCantsLlargs > 1.2) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsLlargs - 1.2) / (1.4-1.2)) : 1)), potencia);
        double valorEsCantLlargIrregular = Math.min(1, 1 - nivellConfiancaEsCantLlargIrregular + 0.5);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 7750) ? Math.max(0, 1.0*(frequenciaMitjana - 6250) / (7750-6250)) : ((frequenciaMitjana > 9250) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 9250) / (10750-9250)) : 1)), potencia);
        double valorPic1Frequencia = Math.pow(((pic1Frequencia < 7000) ? Math.max(0, 1.0*(pic1Frequencia - 6000) / (7000-6000)) : ((pic1Frequencia > 8750) ? Math.max(0, 1 - 1.0*(pic1Frequencia - 8750) / (9750-8750)) : 1)), potencia);
        double valorPic2Frequencia = Math.pow(((pic2Frequencia < 7750) ? Math.max(0, 1.0*(pic2Frequencia - 6750) / (7750-6750)) : ((pic2Frequencia > 9500) ? Math.max(0, 1 - 1.0*(pic2Frequencia - 9500) / (10500-9500)) : 1)), potencia);
        double valorPic3Frequencia = Math.pow(((pic3Frequencia < 8750) ? Math.max(0, 1.0*(pic3Frequencia - 7750) / (8750-7750)) : ((pic3Frequencia > 10250) ? Math.max(0, 1 - 1.0*(pic3Frequencia - 10250) / (11250-10250)) : 1)), potencia);
        double valorNGP = Math.pow(((nombreGrupsDePolsosPerSegon < 66) ? Math.max(0, 1.0*(nombreGrupsDePolsosPerSegon - 63) / (66-63)) : ((nombreGrupsDePolsosPerSegon > 71) ? Math.max(0, 1 - 1.0*(nombreGrupsDePolsosPerSegon - 71) / (74-71)) : 1)), potencia);
        if (nombreGrupsDePolsosPerSegon != ngpAux) {
          valorNGP *= 0.5;
        }
        double valorDefinicioNGP = Math.pow(Math.min(1, Math.max(0, (1 - definicioGrupsDePolsos - 0.2) / (0.45))), potencia);
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(Math.min(1, Math.max(0, (1 - proporcioSilenci - 0.6) / (0.4))), potencia);


        semblancaTibicinaGarricola -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        semblancaTibicinaGarricola -= (1 - valorSonCantsLlargs) * importanciaBaseSonCantsLlargs;
```

```
            semblancaTibicinaGarricola -= (1 - valorPendentPrimeraMeitatCantLlarg) * importanciaBasePendentPrimeraMeitatCantLlarg;
            semblancaTibicinaGarricola -= (1 - valorPendentSegonaMeitatCantLlarg) * importanciaBasePendentSegonaMeitatCantLlarg;
            semblancaTibicinaGarricola -= (1 - valorEsCantLlargIrregular) * importanciaBaseCantLlargIrregular;
            if (fiabilitatIdentificacio != -1) semblancaTibicinaGarricola -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
            if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaGarricola -= (1 - valorPic1Frequencia) * importanciaBasePic1Frequencia;
            if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaGarricola -= (1 - valorPic2Frequencia) * importanciaBasePic2Frequencia;
            if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaGarricola -= (1 - valorPic3Frequencia) * importanciaBasePic3Frequencia;
            semblancaTibicinaGarricola -= (1 - valorNGP) * importanciaBaseNGP;
            semblancaTibicinaGarricola -= (1 - valorDefinicioNGP) * importanciaBaseDefinicioNGP;
            semblancaTibicinaGarricola -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
            semblancaTibicinaGarricola -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

            semblancaTibicinaGarricola = Math.max(0,semblancaTibicinaGarricola);
            return new Pair(semblancaTibicinaGarricola, fiabilitatIdentificacio);
        }
        else if (especie.compareTo("Tibicina haematodes") == 0) {
            /*
            //Tibicina haematodes
            Criteris:
                                                Importància base    0%      100%    100%    0%
            Nivell confiança cants curts amb cant llarg previ                   -1      0,4     0,1     0       -
            Percentatge cants curts                 -0,15       "=100"      "<100"      -
            Nivell confiança cant llarg irregular           -0,3        0,5         0
            Freqüència mitjana                  -0,6        5000    6500    8000        9500
            Pic 1 freqüència                    -0,15       4250    5250    7000        8000
            Pic 2 freqüència                    -0,15       5250    6250    7750        8750
            Pic 3 freqüència                    -0,15       6250    7250    9000        10000
            Nombre de grups de polsos per segon             -1      85      90      105     110
            Definició grups de polsos per segon             -0,1        0,8     0,5     0       -
            Math.max(2, batecs ales / tempsDurada)
                -0,2        2       0       -
            Proporció silencis                  -0,1        1       0,45    0       -

            Mínima frequenciaMaximaMostreig = 9000
            Mínima frequenciaMaximaMostreig pels 3 pics = 10000
            Confiança freqüència = (4-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),6))/4
            freqüència += 500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),6)
            */

            double semblancaTibicinaHaematodes = 1.0;

            double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
            double importanciaBasePercentatgeCantsCurts = 0.15;
            double importanciaBaseCantLlargIrregular = 0.3;
            double importanciaBaseMitjanaFrequencia = 0.6;
            double importanciaBasePic1Frequencia = 0.15;
            double importanciaBasePic2Frequencia = 0.15;
            double importanciaBasePic3Frequencia = 0.15;
            double importanciaBaseNGP = 1.0;
            double importanciaBaseDefinicioNGP = 0.1;
            double importanciaBaseBatecsAles = 0.2;
            double importanciaBaseProporcioSilenci = 0.1;


            frequenciaMitjana += 500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),6);

            double fiabilitatIdentificacio;
            if (frequenciaMaximaMostreig < 9000) fiabilitatIdentificacio = -1.0;
            else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
            else fiabilitatIdentificacio = 1.0;


            double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
            double valorPercentatgeCantsCurts = percentatgeCantsCurts != 100 ? 1 : 0;
            double valorEsCantLlargIrregular = Math.min(1, 1 - nivellConfiancaEsCantLlargIrregular + 0.5);
            double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 6500) ? Math.max(0, 1.0*(frequenciaMitjana - 5000) / (6500-5000)) : ((frequenciaMitjana > 8000) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 8000) / (9500-8000)) : 1)), potencia);
            double valorPic1Frequencia = Math.pow(((pic1Frequencia < 5250) ? Math.max(0, 1.0*(pic1Frequencia - 4250) / (5250-4250)) : ((pic1Frequencia > 7000) ? Math.max(0, 1 - 1.0*(pic1Frequencia - 7000) / (8000-7000)) : 1)), potencia);
            double valorPic2Frequencia = Math.pow(((pic2Frequencia < 6250) ? Math.max(0, 1.0*(pic2Frequencia - 5250) / (6750-5250)) : ((pic2Frequencia > 7750) ? Math.max(0, 1 - 1.0*(pic2Frequencia - 7750) / (8750-7750)) : 1)), potencia);
            double valorPic3Frequencia = Math.pow(((pic3Frequencia < 7250) ? Math.max(0, 1.0*(pic3Frequencia - 6250) / (7250-6250)) : ((pic3Frequencia > 9000) ? Math.max(0, 1 - 1.0*(pic3Frequencia - 9000) / (10000-9000)) : 1)), potencia);
            double valorNGP = Math.pow(((nombreGrupsDePolsosPerSegon < 90) ? Math.max(0, 1.0*(nombreGrupsDePolsosPerSegon - 85) / (90-85)) : ((nombreGrupsDePolsosPerSegon > 105) ? Math.max(0, 1 - 1.0*(nombreGrupsDePolsosPerSegon - 105) / (110-105)) : 1)), potencia);
            double valorDefinicioNGP = Math.pow(Math.min(1, Math.max(0, (1 - definicioGrupsDePolsos - 0.2) / (0.3))), potencia);
            double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
            double valorProporcioSilenci = Math.pow(Math.min(1, Math.max(0, (1 - proporcioSilenci - 0.0) / (0.55))), potencia);


            semblancaTibicinaHaematodes -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
            semblancaTibicinaHaematodes -= (1 - valorPercentatgeCantsCurts) * importanciaBasePercentatgeCantsCurts;
            semblancaTibicinaHaematodes -= (1 - valorEsCantLlargIrregular) * importanciaBaseCantLlargIrregular;
            if (fiabilitatIdentificacio != -1) semblancaTibicinaHaematodes -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
            if (frequenciaMaximaMostreig >= 10000) semblancaTibicinaHaematodes -= (1 - valorPic1Frequencia) * importanciaBasePic1Frequencia;
            if (frequenciaMaximaMostreig >= 10000) semblancaTibicinaHaematodes -= (1 - valorPic2Frequencia) * importanciaBasePic2Frequencia;
            if (frequenciaMaximaMostreig >= 10000) semblancaTibicinaHaematodes -= (1 - valorPic3Frequencia) * importanciaBasePic3Frequencia;
            semblancaTibicinaHaematodes -= (1 - valorNGP) * importanciaBaseNGP;
            semblancaTibicinaHaematodes -= (1 - valorDefinicioNGP) * importanciaBaseDefinicioNGP;
            semblancaTibicinaHaematodes -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
            semblancaTibicinaHaematodes -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

            semblancaTibicinaHaematodes = Math.max(0,semblancaTibicinaHaematodes);
            return new Pair(semblancaTibicinaHaematodes, fiabilitatIdentificacio);
        }
        else if (especie.compareTo("Tibicina quadrisignata") == 0) {
            /*
            //Tibicina quadrisignata
            Criteris:
                                                Importància base    0%      100%    100%    0%
            Durada mitjana cant curt                -0,15       100     200     700     -
            Nivell confiança cants curts amb cant llarg previ               -1      0,4     0,1     0       -
            Pendent primera meitat cant curt                -0,05       0,85        1       1,5     2
            Pendent segona meitat cant curt             -0,05       0,8     0,9     2       2,5
            Pendent primera meitat cant llarg           -0,05       0,8     0,9     1,5     2
            Pendent segona meitat cant llarg            -0,05       0,8     0,9     2       2,5
            Nivell confiança cant llarg irregular           -0,2        0,5         0
            Freqüència mitjana                  -1,0        6500    8000    9500        11000
            Pic 1 freqüència                    -0,15       5750    7250    8500        10000
            Pic 2 freqüència                    -0,15       6250    7750    9250        10750
            Pic 3 freqüència                    -0,15       7250    8750    10250       11750
            Nombre de grups de polsos per segon             -1      67      73      80      85
            Definició grups de polsos per segon             -0,1        0,8     0,4     0       -
            Math.max(2, batecs ales / tempsDurada)
                -0,2        2       0       -
            Proporció silencis                  -0,1        1       0,45    0       -

            Mínima frequenciaMaximaMostreig = 9000
            Mínima frequenciaMaximaMostreig pels 3 pics = 11000
```

```
        Confiança freqüència = (4-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),8))/4
        freqüència += 1500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),8)
        */

        double semblancaTibicinaQuadrisignata = 1.0;

        double importanciaBaseMitjanaDuradaCantCurt = 0.15;
        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
        double importanciaBasePendentPrimeraMeitatCantCurt = 0.05;
        double importanciaBasePendentSegonaMeitatCantCurt = 0.05;
        double importanciaBasePendentPrimeraMeitatCantLlarg = 0.05;
        double importanciaBasePendentSegonaMeitatCantLlarg = 0.05;
        double importanciaBaseCantLlargIrregular = 0.2;
        double importanciaBaseMitjanaFrequencia = 1.0;
        double importanciaBasePic1Frequencia = 0.15;
        double importanciaBasePic2Frequencia = 0.15;
        double importanciaBasePic3Frequencia = 0.15;
        double importanciaBaseNGP = 1.0;
        double importanciaBaseDefinicioNGP = 0.1;
        double importanciaBaseBatecsAles = 0.2;
        double importanciaBaseProporcioSilenci = 0.1;


        frequenciaMitjana += 1500*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),8);

        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 9000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;


        double valorMitjanaDuradaCantCurt = Math.pow(Math.min(1, Math.max(0, (duradaMitjanaCantCurt - 100) / (100))), potencia);
        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorPendentPrimeraMeitatCantCurt = Math.pow(((pendentPrimeraMeitatCantsCurts < 1.0) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsCurts - 0.85) / (1.0-0.85)) : ((pendentPrimeraMeitatCantsCurts > 1.5) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsCurts - 1.5) / (2.0-1.5)) : 1)), potencia);
        double valorPendentSegonaMeitatCantCurt = Math.pow(((pendentSegonaMeitatCantsCurts < 0.9) ? Math.max(0, 1.0*(pendentSegonaMeitatCantsCurts - 0.8) / (0.9-0.8)) : ((pendentSegonaMeitatCantsCurts > 2.0) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsCurts - 2.0) / (2.5-2.0)) : 1)), potencia);
        double valorPendentPrimeraMeitatCantLlarg = Math.pow(((pendentPrimeraMeitatCantsLlargs < 0.9) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsLlargs - 0.8) / (0.9-0.8)) : ((pendentPrimeraMeitatCantsLlargs > 1.5) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsLlargs - 1.5) / (2.0-1.5)) : 1)), potencia);
        double valorPendentSegonaMeitatCantLlarg = Math.pow(((pendentSegonaMeitatCantsLlargs < 0.9) ? Math.max(0, 1.0*(pendentSegonaMeitatCantsLlargs - 0.8) / (0.9-0.8)) : ((pendentSegonaMeitatCantsLlargs > 2.0) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsLlargs - 2.0) / (2.5-2.0)) : 1)), potencia);
        double valorEsCantLlargIrregular = Math.min(1, 1 - nivellConfiancaEsCantLlargIrregular + 0.5);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 8000) ? Math.max(0, 1.0*(frequenciaMitjana - 6500) / (8000-6500)) : ((frequenciaMitjana > 9500) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 9500) / (11000-9500)) : 1)), potencia);
        double valorPic1Frequencia = Math.pow(((pic1Frequencia < 7250) ? Math.max(0, 1.0*(pic1Frequencia - 5750) / (7250-5750)) : ((pic1Frequencia > 8500) ? Math.max(0, 1 - 1.0*(pic1Frequencia - 8500) / (10000-8500)) : 1)), potencia);
        double valorPic2Frequencia = Math.pow(((pic2Frequencia < 7750) ? Math.max(0, 1.0*(pic2Frequencia - 6250) / (7750-6250)) : ((pic2Frequencia > 9250) ? Math.max(0, 1 - 1.0*(pic2Frequencia - 9250) / (10750-9250)) : 1)), potencia);
        double valorPic3Frequencia = Math.pow(((pic3Frequencia < 8750) ? Math.max(0, 1.0*(pic3Frequencia - 7250) / (8750-7250)) : ((pic3Frequencia > 10250) ? Math.max(0, 1 - 1.0*(pic3Frequencia - 10250) / (11750-10250)) : 1)), potencia);
        double valorNGP = Math.pow(((nombreGrupsDePolsosPerSegon < 73) ? Math.max(0, 1.0*(nombreGrupsDePolsosPerSegon - 67) / (73-67)) : ((nombreGrupsDePolsosPerSegon > 80) ? Math.max(0, 1 - 1.0*(nombreGrupsDePolsosPerSegon - 80) / (85-80)) : 1)), potencia);
        double valorDefinicioNGP = Math.pow(Math.min(1, Math.max(0, (1 - definicioGrupsDePolsos - 0.2) / (0.4))), potencia);
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(Math.min(1, Math.max(0, (1 - proporcioSilenci - 0.0) / (0.55))), potencia);


        semblancaTibicinaQuadrisignata -= (1 - valorMitjanaDuradaCantCurt) * importanciaBaseMitjanaDuradaCantCurt;
        semblancaTibicinaQuadrisignata -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        semblancaTibicinaQuadrisignata -= (1 - valorPendentPrimeraMeitatCantCurt) * importanciaBasePendentPrimeraMeitatCantCurt;
        semblancaTibicinaQuadrisignata -= (1 - valorPendentSegonaMeitatCantCurt) * importanciaBasePendentSegonaMeitatCantCurt;
        semblancaTibicinaQuadrisignata -= (1 - valorPendentPrimeraMeitatCantLlarg) * importanciaBasePendentPrimeraMeitatCantLlarg;
        semblancaTibicinaQuadrisignata -= (1 - valorPendentSegonaMeitatCantLlarg) * importanciaBasePendentSegonaMeitatCantLlarg;
        semblancaTibicinaQuadrisignata -= (1 - valorEsCantLlargIrregular) * importanciaBaseCantLlargIrregular;
        if (fiabilitatIdentificacio != -1) semblancaTibicinaQuadrisignata -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaQuadrisignata -= (1 - valorPic1Frequencia) * importanciaBasePic1Frequencia;
        if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaQuadrisignata -= (1 - valorPic2Frequencia) * importanciaBasePic2Frequencia;
        if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaQuadrisignata -= (1 - valorPic3Frequencia) * importanciaBasePic3Frequencia;
        semblancaTibicinaQuadrisignata -= (1 - valorNGP) * importanciaBaseNGP;
        semblancaTibicinaQuadrisignata -= (1 - valorDefinicioNGP) * importanciaBaseDefinicioNGP;
        semblancaTibicinaQuadrisignata -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaTibicinaQuadrisignata -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaTibicinaQuadrisignata = Math.max(0,semblancaTibicinaQuadrisignata);
        return new Pair(semblancaTibicinaQuadrisignata, fiabilitatIdentificacio);
      }
      else if (especie.compareTo("Tibicina tomentosa") == 0) {
        /*
        //Tibicina tomentosa
        Criteris:
                                        Importància base    0%      100%    100%    0%
        Nivell confiança cants curts amb cant llarg previ       -1      0,4     0,1     0       -
        Percentatge cants curts         -0,15   "=100"  "<100"  -
        Pendent primera meitat cant llarg       -0,15   0,8     0,95    1,3     1,65
        Pendent segona meitat cant llarg        -0,15   0,8     0,95    1,2     1,55
        Nivell confiança cant llarg irregular   -0,2    0,5     0       -
        Freqüència mitjana      -1      6500    7750    9250    10500
        Pic 1 freqüència        -0,2    5500    7000    8500    10000
        Pic 2 freqüència        -0,2    6250    7750    9500    11000
        Pic 3 freqüència        -0,2    7500    9000    10500   12000
        Nombre de grups de polsos per segon     -1      138     145     180     185
        Definició grups de polsos per segon     -0,15   0       0,15    1       -
        Math.max(2, batecs ales / tempsDurada)
        -0,2    2       0       -
        Proporció silencis      -0,15   1       0,3     0       -

        Mínima frequenciaMaximaMostreig = 9000
        Mínima frequenciaMaximaMostreig pels 3 pics = 11000
        Confiança freqüència = (4-Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),8))/4
        freqüència += 1200*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),8)
        */

        double semblancaTibicinaTomentosa = 1.0;

        double importanciaBaseCantsCurtsAmbCantLlargPrevi = 1.0;
        double importanciaBasePercentatgeCantsCurts = 0.15;
        double importanciaBasePendentPrimeraMeitatCantLlarg = 0.15;
        double importanciaBasePendentSegonaMeitatCantLlarg = 0.15;
        double importanciaBaseCantLlargIrregular = 0.2;
        double valorEsCantLlargIrregular = Math.min(1, 1 - nivellConfiancaEsCantLlargIrregular + 0.5);
        double importanciaBaseMitjanaFrequencia = 1.0;
        double importanciaBasePic1Frequencia = 0.2;
        double importanciaBasePic2Frequencia = 0.2;
        double importanciaBasePic3Frequencia = 0.2;
        double importanciaBaseNGP = 1.0;
        double importanciaBaseDefinicioNGP = 0.15;
        double importanciaBaseBatecsAles = 0.2;
        double importanciaBaseProporcioSilenci = 0.15;


        frequenciaMitjana += 1200*Math.pow((1-(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),8);
```

```
        double fiabilitatIdentificacio;
        if (frequenciaMaximaMostreig < 9000) fiabilitatIdentificacio = -1.0;
        else if ((mitjanaAmplitud*1+mitjanaAmplitudCant*4)/5 < 200) fiabilitatIdentificacio = -2.0;
        else fiabilitatIdentificacio = 1.0;


        double valorCantsCurtsAmbCantLlargPrevi = Math.pow(Math.min(1, Math.max(0, (1 - nivellConfiancaCantsCurtsAmbCantLlargPrevi - 0.6) / (0.3))), potencia);
        double valorPercentatgeCantsCurts = percentatgeCantsCurts != 100 ? 1 : 0;
        double valorPendentPrimeraMeitatCantLlarg = Math.pow(((pendentPrimeraMeitatCantsLlargs < 0.95) ? Math.max(0, 1.0*(pendentPrimeraMeitatCantsLlargs - 0.8) / (0.95-0.8)) : ((pendentPrimeraMeitatCantsLlargs > 1.3) ? Math.max(0, 1 - 1.0*(pendentPrimeraMeitatCantsLlargs - 1.3) / (1.65-1.3)) : 1)), potencia);
        double valorPendentSegonaMeitatCantLlarg = Math.pow(((pendentSegonaMeitatCantsLlargs < 0.95) ? Math.max(0, 1.0*(pendentSegonaMeitatCantsLlargs - 0.8) / (0.95-0.8)) : ((pendentSegonaMeitatCantsLlargs > 1.2) ? Math.max(0, 1 - 1.0*(pendentSegonaMeitatCantsLlargs - 1.2) / (1.55-1.2)) : 1)), potencia);
        double valorMitjanaFrequencia = Math.pow(((frequenciaMitjana < 7750) ? Math.max(0, 1.0*(frequenciaMitjana - 6500) / (7750-6500)) : ((frequenciaMitjana > 9250) ? Math.max(0, 1 - 1.0*(frequenciaMitjana - 9250) / (10500-9250)) : 1)), potencia);
        double valorPic1Frequencia = Math.pow(((pic1Frequencia < 7000) ? Math.max(0, 1.0*(pic1Frequencia - 5500) / (7000-5500)) : ((pic1Frequencia > 8500) ? Math.max(0, 1 - 1.0*(pic1Frequencia - 8500) / (10000-8500)) : 1)), potencia);
        double valorPic2Frequencia = Math.pow(((pic2Frequencia < 7750) ? Math.max(0, 1.0*(pic2Frequencia - 6250) / (7750-6250)) : ((pic2Frequencia > 9500) ? Math.max(0, 1 - 1.0*(pic2Frequencia - 9500) / (11000-9500)) : 1)), potencia);
        double valorPic3Frequencia = Math.pow(((pic3Frequencia < 9000) ? Math.max(0, 1.0*(pic3Frequencia - 7500) / (9000-7500)) : ((pic3Frequencia > 10500) ? Math.max(0, 1 - 1.0*(pic3Frequencia - 10500) / (12000-10500)) : 1)), potencia);
        double valorNGP = Math.pow(((nombreGrupsDePolsosPerSegon < 145) ? Math.max(0, 1.0*(nombreGrupsDePolsosPerSegon - 138) / (145-138)) : ((nombreGrupsDePolsosPerSegon > 180) ? Math.max(0, 1 - 1.0*(nombreGrupsDePolsosPerSegon - 180) / (185-180)) : 1)), potencia);
        double valorDefinicioNGP = Math.pow(Math.min(1, Math.max(0, (definicioGrupsDePolsos) / (0.15))), potencia);
        double valorBatecsAles = 2 - Math.min(1, 2.0*nBatecsAlesOCantsIncorrectes / tempsDurada);
        double valorProporcioSilenci = Math.pow(Math.min(1, Math.max(0, (1 - proporcioSilenci - 0.0) / (0.7))), potencia);


        semblancaTibicinaTomentosa -= (1 - valorCantsCurtsAmbCantLlargPrevi) * importanciaBaseCantsCurtsAmbCantLlargPrevi;
        semblancaTibicinaTomentosa -= (1 - valorPercentatgeCantsCurts) * importanciaBasePercentatgeCantsCurts;
        semblancaTibicinaTomentosa -= (1 - valorPendentPrimeraMeitatCantLlarg) * importanciaBasePendentPrimeraMeitatCantLlarg;
        semblancaTibicinaTomentosa -= (1 - valorPendentSegonaMeitatCantLlarg) * importanciaBasePendentSegonaMeitatCantLlarg;
        semblancaTibicinaTomentosa -= (1 - valorEsCantLlargIrregular) * importanciaBaseCantLlargIrregular;
        if (fiabilitatIdentificacio != -1) semblancaTibicinaTomentosa -= (1 - valorMitjanaFrequencia) * importanciaBaseMitjanaFrequencia;
        if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaTomentosa -= (1 - valorPic1Frequencia) * importanciaBasePic1Frequencia;
        if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaTomentosa -= (1 - valorPic2Frequencia) * importanciaBasePic2Frequencia;
        if (frequenciaMaximaMostreig >= 11000) semblancaTibicinaTomentosa -= (1 - valorPic3Frequencia) * importanciaBasePic3Frequencia;
        semblancaTibicinaTomentosa -= (1 - valorNGP) * importanciaBaseNGP;
        semblancaTibicinaTomentosa -= (1 - valorDefinicioNGP) * importanciaBaseDefinicioNGP;
        semblancaTibicinaTomentosa -= (2 - valorBatecsAles) * importanciaBaseBatecsAles;
        semblancaTibicinaTomentosa -= (1 - valorProporcioSilenci) * importanciaBaseProporcioSilenci;

        semblancaTibicinaTomentosa = Math.max(0,semblancaTibicinaTomentosa);
        return new Pair(semblancaTibicinaTomentosa, fiabilitatIdentificacio);
      }
      else return new Pair(0.0, 0.0);
  }

  /**
    @throws OutOfMemoryError
    @pre ---
    @post Retorna els resultats d'identificació corresponents al fitxer fitxer.
    @return Els resultats d'identificació corresponents al fitxer fitxer
    @param resultatsIdentificacio String que conté el nom del fitxer en qüestió per afegir-hi posteriorment els resultats obtinguts
  */
  private static String analitzaFitxer(File fitxer, String resultatsIdentificacio) throws OutOfMemoryError, Exception {
    try {
      //Llegeix les dades d'àudio del fitxer.
      DadesAudio dadesAudio = obtinguesDadesAudio(new BufferedInputStream(new FileInputStream(fitxer)));
      byte[] bytes = new byte[(int) (dadesAudio.obtinguesLlargadaEnQuadres()) * (dadesAudio.obtinguesFormat().obtinguesMidaDeQuadre())];
      dadesAudio.read(bytes);

      int fs = Math.round(dadesAudio.obtinguesFormat().obtinguesFrequenciaDeMostreig())*dadesAudio.obtinguesFormat().obtinguesMidaDeQuadre()/4;

      //Obté els valors d'amplitud per a cada canal d'àudio.
      int[][] valorsAmplitud = obtinguesAmplitud(bytes, 1);

      if (valorsAmplitud[0].length > 4500000) return "Atesa la durada de la gravació, la memòria requerida per a la seva anàlisi és massa elevada. El recomanable és realitzar enregistraments d'uns deu o quinze segons.\n";

      FormatWav format = dadesAudio.obtinguesFormat();

      //Quadre = "frame" en anglès
      long quadres = dadesAudio.obtinguesLlargadaEnQuadres();
      double duradaEnSegons = (quadres+0.0) / format.obtinguesQuadresPerSegon();

      //Emmagatzema els valors d'amplitud del primer canal d'àudio en una ArrayList, amb la qual serà més pràctic treballar.
      ArrayList<Integer> valorsAmplitudMono = new ArrayList<>();

      for(int i = 0; i < valorsAmplitud[0].length; i++) {
        valorsAmplitudMono.add(valorsAmplitud[0][i]);
      }


      //Elimina les dades que ja no necessita per alliberar massa memòria.
      valorsAmplitud = null;
      bytes = null;
      format = null;
      dadesAudio = null;
      System.gc();

      //Prepara els valors d'amplitud per realitzar la transformada de Fourier.
      double[] x = deRealsARealsPerAFFT2(valorsAmplitudMono);
      double[] y = new double[x.length];

      for(int i = 0; i < y.length; i++) {
        y[i] = 0.0;
      }

      //Calcula la transformada ràpida de Fourier i posteriorment l'escala per necessitar menys memòria i menys temps per recórrer-la.
      FFT fft = new FFT(y.length);
      fft.fft(x, y);

      fft = null;
      System.gc();

      //Escalarà els valors obtinguts a 1024, amb la qual cosa els valors en què s'ha de fixar (amplitud respecte a la freqüència) són els primers 256 (la primera quarta part).
      int midaFinestra = 1024;

      ArrayList<Double> fftReduida = realsRedueixA(x, y, midaFinestra);

      x = null;
      y = null;
      System.gc();

      ArrayList<Double> fftReduidaPrimeraPart = new ArrayList<>();
      for(int i = 0; i < midaFinestra*0.25; i++) {
        fftReduidaPrimeraPart.add(fftReduida.get(i));
      }

      //Troba els pics de la gràfica d'amplitud respecte a la freqüència.
      ArrayList<Integer> pics = trobaTresPics(fftReduidaPrimeraPart, fs);

      //Determina la freqüència de mostreig real, que no sempre coincideix amb la teòrica.
      int fsReal = trobaFsReal(fftReduidaPrimeraPart, fs);

      //Filtra el soroll del fitxer aplicant un filtre a partir dels coeficients band-pass de Butterworth.
      Pair<Double, Double> frequenciesTall = trobaFrequenciesTallAbansDeReduir(fftReduidaPrimeraPart, fs);

      double freqTall = frequenciesTall.getFirst();
      double freqTallSuperior = frequenciesTall.getSecond();

      double mitjFreq = trobaMitjanaFrequencia(fftReduidaPrimeraPart, fs, freqTall, freqTallSuperior);

      ArrayList<Double> amplitudsModificades = filtraPerBandPass(valorsAmplitudMono, freqTall, freqTallSuperior);
```

```
        //Amplifica el so tant com es pot.
        int[][] amplitudsFiltrades = new int[1][amplitudsModificades.size()];

        for (int i = 0; i < amplitudsModificades.size(); i++) {
            amplitudsFiltrades[0][i] = (int) Math.round(amplitudsModificades.get(i));
        }

        byte[] bytesFiltratsAmpliats = amplificaAmplituds(obtinguesBytes(amplitudsFiltrades, 1));

        valorsAmplitud = obtinguesAmplitud(bytesFiltratsAmpliats, 1);

        amplitudsModificades = new ArrayList<>();

        for(int i = 0; i < valorsAmplitud[0].length; i++) {
            amplitudsModificades.add(valorsAmplitud[0][i]*1.0);
        }

        valorsAmplitud = null;
        valorsAmplitudMono = null;
        System.gc();

        //Passa les amplituds a valor absolut per poder treballar-hi més fàcilment.
        ArrayList<Double> amplitudsModificadesAbsolutes = passaAValorAbsolut(amplitudsModificades);

        //Cerca a quins trams pot haver-hi un cant
        ArrayList<Pair<Double, Double> > llistaCants = trobaLlistaCants(amplitudsModificades, amplitudsModificadesAbsolutes, fs, (int) Math.round(0.005*fs));

        boolean hiHaAlgunCantPerAnalitzar = false;
        boolean amplitudsReduides025msCalculades = false;
        int iteradorAuxiliar = 0;

        while (!hiHaAlgunCantPerAnalitzar && iteradorAuxiliar < llistaCants.size()) {
            if (llistaCants.get(iteradorAuxiliar).getSecond() - llistaCants.get(iteradorAuxiliar).getFirst() > 500) hiHaAlgunCantPerAnalitzar = true;
            iteradorAuxiliar++;
        }

        //Agrupa les amplituds en grups de 0.25 ms.
        ArrayList<Double> amplitudsModificadesAbsolutesReduides025ms = new ArrayList<>();
        int pas = (int) Math.round(2*fs*0.00025);

        //Si no hi ha cap cant per analitzar, no val la pena que calculi les amplituds reduïdes.
        if (hiHaAlgunCantPerAnalitzar) {
            for(int i = 0; i < amplitudsModificadesAbsolutes.size() - pas; i += pas) {
                amplitudsModificadesAbsolutesReduides025ms.add(trobaMitjanaAmbValorsAbsoluts(i, pas, amplitudsModificadesAbsolutes));
            }
            amplitudsReduides025msCalculades = true;
        }

        //Calcula màxims i mínims de les durades dels cants i els silencis.
        double minimaDuradaCant = Double.MAX_VALUE;
        double maximaDuradaCant = 0;

        double minimaDuradaSilenci = Double.MAX_VALUE;
        double maximaDuradaSilenci = 0;

        for (Pair<Double, Double> llistaCant : llistaCants) {
            if (llistaCant.getSecond() - llistaCant.getFirst() > maximaDuradaCant) {
                maximaDuradaCant = llistaCant.getSecond() - llistaCant.getFirst();
            }
            if (llistaCant.getSecond() - llistaCant.getFirst() < minimaDuradaCant) {
                minimaDuradaCant = llistaCant.getSecond() - llistaCant.getFirst();
            }
        }

        for(int i = 1; i < llistaCants.size(); i++) {
            if (llistaCants.get(i).getFirst() - llistaCants.get(i-1).getSecond() > maximaDuradaSilenci) maximaDuradaSilenci = llistaCants.get(i).getFirst() - llistaCants.get(i-1).getSecond();
            if (llistaCants.get(i).getFirst() - llistaCants.get(i-1).getSecond() < minimaDuradaSilenci) minimaDuradaSilenci = llistaCants.get(i).getFirst() - llistaCants.get(i-1).getSecond();
        }

        //Escala les amplituds ajuntant-les en grups d'1 ms. Interessarà un agrupament més o menys precís en funció de l'anàlisi per a la qual s'utilitzi els valors d'amplitud.
        ArrayList<Double> amplitudsModificadesAbsolutesReduides = redueixA(amplitudsModificadesAbsolutes, amplitudsModificadesAbsolutes.size()/(fs*2/1000));

        //Si els cants estan separats per un espai molt curt, els ajunta.
        int silencisMoltCurts = 0;
        int it = 0;
        while (silencisMoltCurts < 2 && it < llistaCants.size()-1) {
            if (llistaCants.get(it+1).getFirst() - llistaCants.get(it).getSecond() < 35) silencisMoltCurts++;
            it++;
        }

        if (silencisMoltCurts > 3 && silencisMoltCurts / llistaCants.size() > 0.25) {
            double inici = llistaCants.get(0).getFirst();
            double fi = llistaCants.get(llistaCants.size()-1).getSecond();
            llistaCants.clear();
            llistaCants.add(new Pair(inici, fi));
        }

        //Assigna els coeficients que utilitzaré per determinar els inicis i finals concrets de cada iteració de cant.
        int duradaMinSilenci = 28;
        double coef = 2.5;
        double coef2 = 1.75;
        double coef3 = 1.75;

        ArrayList<Pair<Double, Double> > llistaCantsDetallada = new ArrayList< >();

        //De cada cant, n'obté les iteracions de cant concretes.
        for (Pair<Double, Double> llistaCant : llistaCants) {
            if (llistaCant.getSecond() - llistaCant.getFirst() > 500) {
                llistaCantsDetallada.addAll(trobaCantsConcrets(amplitudsModificadesAbsolutesReduides025ms, fs, llistaCant.getFirst(), llistaCant.getSecond(), duradaMinSilenci, coef, coef2, coef3));
            } else {
                llistaCantsDetallada.add(llistaCant);
            }
        }

        ArrayList<Integer> indexsCantsContinusPlebejus = new ArrayList<>();

        //Obté els cants continus de la Lyristes plebejus (els ha marcat assignant un valor negatiu al milisegon d'inici del cant).
        for(int i = 0; i < llistaCantsDetallada.size(); i++) {
            if (llistaCantsDetallada.get(i).getFirst() < 0) {
                indexsCantsContinusPlebejus.add(i);
                llistaCantsDetallada.set(i, new Pair(-llistaCantsDetallada.get(i).getFirst(), llistaCantsDetallada.get(i).getSecond()));
            }
        }

        //Determina les durades total i mitjana dels cants trobats.
        double duradaTotalCants = 0;
        for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
            duradaTotalCants += llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
        }

        double duradaMitjanaCant = duradaTotalCants / llistaCantsDetallada.size();

        //Per analitzar els cants de la Lyristes plebejus s'utilitza una precisió diferent, més adequada al seu cant característic.
        ArrayList<Pair<Double, Double> > llistaAbansAnalisiLyristes = new ArrayList<>();
        llistaAbansAnalisiLyristes.addAll(llistaCantsDetallada);

        //Intenta evitar que tregui cants curts amb cant llarg previ en cas que n'hi hagi. Per això guarda els que té per a més endavant.
        ArrayList<Pair<Double, Double> > cantsCurtsAmbCantLlargPreviAuxiliar = new ArrayList<>();
        ArrayList<Pair<Double, Double> > cantsLlargsAmbCantCurtPosteriorAuxiliar = new ArrayList<>();

        for(int i = 1; i < llistaCantsDetallada.size(); i++) {
            if (sonCantLlargICantCurt(llistaCantsDetallada, i)) {
                cantsCurtsAmbCantLlargPreviAuxiliar.add(llistaCantsDetallada.get(i));
                cantsLlargsAmbCantCurtPosteriorAuxiliar.add(llistaCantsDetallada.get(i-1));
            }
        }

        //De vegades, la Tettigettula pygmea fa un cant llarg que enllaça amb un cant curt, i comprova que no sigui el cas.
        double esCantLlargICantCurtPosteriorTettigettulaPygmea;

        if (cantsCurtsAmbCantLlargPreviAuxiliar.isEmpty()) esCantLlargICantCurtPosteriorTettigettulaPygmea = 0.0;
        else {
            esCantLlargICantCurtPosteriorTettigettulaPygmea = 1.0;
            double mitjanaCantCurtTetPyg = 0.0;
            double mitjanaCantLlargTetPyg = 0.0;
            double mitjanaSilenciEntreCantsLlargICurtPosteriorTetPyg = 0.0;

            for(int i = 0; i < cantsLlargsAmbCantCurtPosteriorAuxiliar.size(); i++) {
```

```
            mitjanaCantLlargTetPyg += cantsLlargsAmbCantCurtPosteriorAuxiliar.get(i).getSecond() - cantsLlargsAmbCantCurtPosteriorAuxiliar.get(i).getFirst();
            mitjanaCantCurtTetPyg += cantsCurtsAmbCantLlargPreviAuxiliar.get(i).getSecond() - cantsCurtsAmbCantLlargPreviAuxiliar.get(i).getFirst();
            mitjanaSilenciEntreCantsLlarglCurtPosteriorTetPyg += cantsCurtsAmbCantLlargPreviAuxiliar.get(i).getFirst() - cantsLlargsAmbCantCurtPosteriorAuxiliar.get(i).getSecond();
        }
        mitjanaCantCurtTetPyg /= cantsLlargsAmbCantCurtPosteriorAuxiliar.size();
        mitjanaCantLlargTetPyg /= cantsLlargsAmbCantCurtPosteriorAuxiliar.size();
        mitjanaSilenciEntreCantsLlarglCurtPosteriorTetPyg /= cantsLlargsAmbCantCurtPosteriorAuxiliar.size();

        esCantLlarglCantCurtPosteriorTettigettulaPygmea -= Math.min(1,Math.max(0,(40 - mitjanaSilenciEntreCantsLlarglCurtPosteriorTetPyg) / 40));
        esCantLlarglCantCurtPosteriorTettigettulaPygmea -= Math.min(1,Math.max(0, Math.abs((450 - mitjanaCantLlargTetPyg)/450)));
        esCantLlarglCantCurtPosteriorTettigettulaPygmea -= Math.min(1,Math.max(0, Math.abs((60 - mitjanaCantCurtTetPyg)/60)));
        esCantLlarglCantCurtPosteriorTettigettulaPygmea *= Math.sqrt(Math.sqrt(Math.min(1,Math.max(0, Math.log10(cantsLlargsAmbCantCurtPosteriorAuxiliar.size())))));
    }


    //Lyristes plebejus, Cicada barbara lusitanica, Cicadetta brevipennis o Tibicina
    if (duradaMitjanaCant > 500 || esCantLlarglCantCurtPosteriorTettigettulaPygmea > 0.4) {
        //Durada i coeficients adaptats als cants llargs perquè, en cas que es tracti d'aquest tipus de cants, en detecti bé l'inici i el final
        duradaMinSilenci = 10;
        coef = 2.75;
        coef2 = 1.75;
        coef3 = 2.25;
        llistaCantsDetallada.clear();

        for (Pair<Double, Double> llistaCant : llistaCants) {
            if (llistaCant.getSecond() - llistaCant.getFirst() > 500) {

llistaCantsDetallada.addAll(trobaCantsConcrets(amplitudsModificadesAbsolutesRedui des025ms, fs, llistaCant.getFirst(), llistaCant.getSecond(), duradaMinSilenci, coef, coef2, coef3));
            } else {
                llistaCantsDetallada.add(llistaCant);
            }
        }

        //Comprova quins fragments s'ha eliminat respecte a l'anàlisi convencional i determina si estan ben eliminats o no. En cas contrari, els afegeix de nou.
        ArrayList<Pair<Double, Double> > fragmentsEliminats = new ArrayList<>();
        ArrayList<Pair<Double, Double> > fragmentsConservats = new ArrayList<>();

        if (llistaCantsDetallada.isEmpty()) {
            fragmentsEliminats.addAll(llistaCants);
        }
        else {
            double iniciCantDetall;
            int j;
            for (Pair<Double, Double> llistaCant : llistaCants) {
                j = 0;
                iniciCantDetall = llistaCantsDetallada.get(j).getFirst();
                while (j < llistaCantsDetallada.size() && llistaCant.getFirst() > iniciCantDetall) {
                    j++;
                    iniciCantDetall = llistaCantsDetallada.get(j).getFirst();
                }
                if (iniciCantDetall >= llistaCant.getFirst()) {
                    if (iniciCantDetall > llistaCant.getFirst()) {
                        fragmentsEliminats.add(new Pair(llistaCant.getFirst(), iniciCantDetall));
                    }
                    fragmentsConservats.add(new Pair(iniciCantDetall, llistaCantsDetallada.get(j).getSecond()));
                    while (j < llistaCantsDetallada.size() - 1 && llistaCantsDetallada.get(j).getSecond() < llistaCant.getSecond() && llistaCantsDetallada.get(j+1).getFirst() < llistaCant.getSecond()) {
                        fragmentsEliminats.add(new Pair(llistaCantsDetallada.get(j).getSecond(), llistaCantsDetallada.get(j+1).getFirst()));
                        fragmentsConservats.add(llistaCantsDetallada.get(j+1));
                        j++;
                    }
                    if (llistaCantsDetallada.get(j).getSecond() < llistaCant.getSecond()) {
                        fragmentsEliminats.add(new Pair(llistaCantsDetallada.get(j).getSecond(), llistaCant.getSecond()));
                    }
                } else {
                    fragmentsEliminats.add(new Pair(llistaCant.getFirst(), llistaCant.getSecond()));
                }
            }
        }

        ArrayList<Double> mitjanaAmplitudEliminats = new ArrayList<>();
        double mitjanaAmplitudConservats = 0.0;
        ArrayList<Double> duradaEliminats = new ArrayList<>();
        double totalDuradaConservats = 0.0;

        for (Pair<Double, Double> fragmentsConservat : fragmentsConservats) {
            totalDuradaConservats += fragmentsConservat.getSecond() - fragmentsConservat.getFirst();
            mitjanaAmplitudConservats += (fragmentsConservat.getSecond() - fragmentsConservat.getFirst()) * trobaMitjana((int) Math.round(fragmentsConservat.getFirst() * fs * 2 / 1000.0), (int) Math.round((fragmentsConservat.getSecond() - fragmentsConservat.getFirst()) * fs * 2 / 1000.0), amplitudsModificadesAbsolutes);
        }

        mitjanaAmplitudConservats /= totalDuradaConservats;


        for (Pair<Double, Double> fragmentsEliminat : fragmentsEliminats) {
            duradaEliminats.add(fragmentsEliminat.getSecond() - fragmentsEliminat.getFirst());
            mitjanaAmplitudEliminats.add(trobaMitjana((int) Math.round(fragmentsEliminat.getFirst() * fs * 2 / 1000.0), (int) Math.round(duradaEliminats.get(duradaEliminats.size()-1)*fs*2/1000.0), amplitudsModificadesAbsolutes));
        }

        int indexInsercio;

        for(int i = 0; i < mitjanaAmplitudEliminats.size(); i++) {
            if ((mitjanaAmplitudEliminats.get(i) > mitjanaAmplitudConservats*0.5) && duradaEliminats.get(i) > 500) {
                indexInsercio = trobaPosicioInsercioOrdenada(llistaCantsDetallada, fragmentsEliminats.get(i));
                llistaCantsDetallada.add(indexInsercio, fragmentsEliminats.get(i));
            }
        }

        //Assegura que no hi hagi iteracions de cant que tinguin una baixada brusca d'amplituds i hagin estat considerades com a iteracions separades.
        agrupaCants(llistaCantsDetallada);


        indexsCantsContinusPlebejus = new ArrayList<>();

        //Obté els cants continus de la Lyristes plebejus (els ha marcat assignant un valor negatiu al milisegon d'inici del cant).
        for(int i = 0; i < llistaCantsDetallada.size(); i++) {
            if (llistaCantsDetallada.get(i).getFirst() < 0) {
                indexsCantsContinusPlebejus.add(i);
                llistaCantsDetallada.set(i, new Pair(-llistaCantsDetallada.get(i).getFirst(), llistaCantsDetallada.get(i).getSecond()));
            }
        }
    }
    else {
        //No era una Lyristes, de manera que és millor que es quedi amb la primera anàlisi.
        llistaCantsDetallada.clear();
        llistaCantsDetallada.addAll(llistaAbansAnalisiLyristes);
    }

    //Insereix els cants curts amb cant llarg previ que pugui haver perdut amb la segona anàlisi.
    int itAux = 0;
    for(int i = 0; i < cantsLlargsAmbCantCurtPosteriorAuxiliar.size(); i++) {
        while (itAux < llistaCantsDetallada.size() && llistaCantsDetallada.get(itAux).getSecond() < cantsLlargsAmbCantCurtPosteriorAuxiliar.get(i).getFirst()) {
            itAux++;
        }
        if (itAux == llistaCantsDetallada.size()) {
            llistaCantsDetallada.add(cantsLlargsAmbCantCurtPosteriorAuxiliar.get(i));
            llistaCantsDetallada.add(cantsCurtsAmbCantLlargPreviAuxiliar.get(i));
            itAux += 2;
        }
        else if (itAux == llistaCantsDetallada.size() - 1 || !sonCantLlarglCantCurt(llistaCantsDetallada, itAux+1)) {
            while (itAux < llistaCantsDetallada.size() && llistaCantsDetallada.get(itAux).getFirst() < cantsCurtsAmbCantLlargPreviAuxiliar.get(i).getSecond()) {
                llistaCantsDetallada.remove(itAux);
            }
            llistaCantsDetallada.add(itAux, cantsLlargsAmbCantCurtPosteriorAuxiliar.get(i));
            llistaCantsDetallada.add(itAux+1, cantsCurtsAmbCantLlargPreviAuxiliar.get(i));
```

```
            itAux += 2;
        }
    }

    //Troba la iteració de cant més llarga de totes les que hi ha a la gravació.
    int indexCantMesLlarg = 0;

    for (int i = 0; i < llistaCantsDetallada.size(); i++) {
        if (llistaCantsDetallada.get(i).getSecond() - llistaCantsDetallada.get(i).getFirst() > llistaCantsDetallada.get(indexCantMesLlarg).getSecond() - llistaCantsDetallada.get(indexCantMesLlarg).getFirst()) indexCantMesLlarg = i;
    }

    //Compta els cants curts i, concretament, els que tenen una longitud que pugui correspondre als d'una Lyristes plebejus.
    int nCantsCurtsAux = 0;

    double dist = llistaCantsDetallada.get(0).getSecond() - llistaCantsDetallada.get(0).getFirst();
    double dist2;
    ArrayList<Pair<Double, Double> > llistaCantsAmbLongitudLyristes = new ArrayList<>();

    if (dist > 30 && dist < 120) llistaCantsAmbLongitudLyristes.add(llistaCantsDetallada.get(0));

    for(int i = 1; i < llistaCantsDetallada.size(); i++) {
        dist = llistaCantsDetallada.get(i).getSecond() - llistaCantsDetallada.get(i).getFirst();
        dist2 = llistaCantsDetallada.get(i).getFirst() - llistaCantsDetallada.get(i-1).getSecond();
        if (dist + dist2 > 35 && dist + dist2 < 165 && dist > 20 && dist < 120 && dist2 > 5 && dist2 < 65) llistaCantsAmbLongitudLyristes.add(llistaCantsDetallada.get(i));
        if (dist < 300) nCantsCurtsAux++;
    }

    //Determina si sembla ser una Lyristes plebejus.
    boolean semblaLyristesPlebejus = llistaCantsAmbLongitudLyristes.size() > nCantsCurtsAux*0.9 && nCantsCurtsAux > 10 && trobaDesviacioEstandard(llistaCantsAmbLongitudLyristes) < 0.25;

    //Lyristes plebejus
    if (semblaLyristesPlebejus) {
        //De vegades, amb les Lyristes plebejus se salta el tram continu del cant perquè el cant més discontinu que ve tot seguit té més amplitud. Per tant, s'assegura que no elimini aquest fragment.
        duradaMinSilenci = 10;
        coef = 2.75;
        coef2 = 1.5;
        coef3 = -2.25;
        llistaCantsDetallada.clear();

        //Si no ha calculat les amplituds reduïdes anteriorment, ho fa ara.
        if (!amplitudsReduides025msCalculades) {
            for(int i = 0; i < amplitudsModificadesAbsolutes.size() - pas; i += pas) {
                amplitudsModificadesAbsolutesReduides025ms.add(trobaMitjanaAmbValorsAbsoluts(i, pas, amplitudsModificadesAbsolutes));
            }
            amplitudsReduides025msCalculades = true;
        }

        //Troba les iteracions de cant concretes.
        for (Pair<Double, Double> llistaCant : llistaCants) {
            if (llistaCant.getSecond() - llistaCant.getFirst() > 50) {
                llistaCantsDetallada.addAll(trobaCantsConcrets(amplitudsModificadesAbsolutesReduides025ms, fs, llistaCant.getFirst(), llistaCant.getSecond(), duradaMinSilenci, coef, coef2, coef3));
            } else {
                llistaCantsDetallada.add(llistaCant);
            }
        }

        indexsCantsContinusPlebejus = new ArrayList<>();

        //Obté els cants continus de la Lyristes plebejus (els ha marcat assignant un valor negatiu al milisegon d'inici del cant).
        for(int i = 0; i < llistaCantsDetallada.size(); i++) {
            if (llistaCantsDetallada.get(i).getFirst() < 0) {
                indexsCantsContinusPlebejus.add(i);
                llistaCantsDetallada.set(i, new Pair(-llistaCantsDetallada.get(i).getFirst(), llistaCantsDetallada.get(i).getSecond()));
            }
        }
    }

    //Elimina els cants massa baixos.
    double mitjanaAmplitudCants = 0.0;
    double tempsTotalCant = 0.0;

    for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
        mitjanaAmplitudCants += (llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst()) * trobaMitjanaAmbMaxims((int) Math.round(llistaCantsDetallada1.getFirst()), (int) Math.round(llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst()), amplitudsModificadesAbsolutesReduides, (int) Math.round(llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst()));
        tempsTotalCant += llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
    }

    mitjanaAmplitudCants /= tempsTotalCant;

    for(int i = 0; i < llistaCantsDetallada.size(); i++) {
        if (trobaMitjanaAmbMaxims((int) Math.round(llistaCantsDetallada.get(i).getFirst()), (int) Math.round(llistaCantsDetallada.get(i).getSecond() - llistaCantsDetallada.get(i).getFirst()), amplitudsModificadesAbsolutesReduides, (int) Math.round(llistaCantsDetallada.get(i).getSecond() - llistaCantsDetallada.get(i).getFirst())) < mitjanaAmplitudCants*0.3) {
            llistaCantsDetallada.remove(i);
            i--;
        }
    }

    //Elimina cants molt baixos una segona vegada perquè, en haver eliminat els primers, la mitjana haurà pujat i possiblement en trobarà més.
    mitjanaAmplitudCants = 0.0;
    tempsTotalCant = 0.0;

    for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
        mitjanaAmplitudCants += (llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst()) * trobaMitjanaAmbMaxims((int) Math.round(llistaCantsDetallada1.getFirst()), (int) Math.round(llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst()), amplitudsModificadesAbsolutesReduides, (int) Math.round(llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst()));
        tempsTotalCant += llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
    }

    mitjanaAmplitudCants /= tempsTotalCant;

    for(int i = 0; i < llistaCantsDetallada.size(); i++) {
        if (trobaMitjanaAmbMaxims((int) Math.round(llistaCantsDetallada.get(i).getFirst()), (int) Math.round(llistaCantsDetallada.get(i).getSecond() - llistaCantsDetallada.get(i).getFirst()), amplitudsModificadesAbsolutesReduides, (int) Math.round(llistaCantsDetallada.get(i).getSecond() - llistaCantsDetallada.get(i).getFirst())) < mitjanaAmplitudCants*0.25) {
            llistaCantsDetallada.remove(i);
            i--;
        }
    }

    //Troba les durades mínima, màxima i mitjana dels cants.
    double mitjanaDuradaCants = 0.0;
    minimaDuradaCant = Double.MAX_VALUE;
    maximaDuradaCant = 0.0;
    double duradaAux;
    ArrayList<Double> mitjanesAmplitudsCants = new ArrayList<>();

    for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
        duradaAux = llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
        mitjanaDuradaCants += duradaAux;
        if (duradaAux > maximaDuradaCant) maximaDuradaCant = duradaAux;
        if (duradaAux < minimaDuradaCant) minimaDuradaCant = duradaAux;
        mitjanesAmplitudsCants.add(trobaMitjanaAmbMaxims((int) Math.round(llistaCantsDetallada1.getFirst()), (int) Math.round(llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst()), amplitudsModificadesAbsolutesReduides, (int) Math.round(llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst())));
    }

    mitjanaDuradaCants /= llistaCantsDetallada.size();

    if (!semblaLyristesPlebejus) agrupaCants(llistaCantsDetallada);

    //No interessen cants de Lyristes plebejus, Tibicina sp. o Cicada barbara lusitanica.
    //Millora l'anàlisi per a la Cicadatra i la Hilaphura varipes, que de vegades fan un cant curt creixent (comença amb poca potència i acaba amb més potència).
    if (maximaDuradaCant < 400 && mitjanaDuradaCants < 300) {
        ArrayList<Double> amplitudsModificadesAbsolutesUltraReduides = new ArrayList<>();

        //Agrupa els milisegons de sis en sis per a les mitjanes.
        int msAgrupacio = 6;
        int pas2 = (int) Math.round(2*fs*msAgrupacio/1000);
```

```
        for(int i = 0; i < amplitudsModificadesAbsolutes.size() - pas2; i += pas2) {

amplitudsModificadesAbsolutesUltraReduides.add(trobaMitjanaAmbValorsAbsoluts(i
, pas2, amplitudsModificadesAbsolutes));
        }

        int finalFragment, iniciFragment, finalFragmentOriginal;
        boolean iniciPujada;
        int duradaFragmentOriginal = 16;
        int duradaFragment = duradaFragmentOriginal;
        double pendentPujada = 0.3*duradaFragmentOriginal*msAgrupacio/100.0;
//30% d'increment cada 100 ms.
        for(int i = 0; i < llistaCantsDetallada.size(); i++) {
          finalFragment = (int)
(Math.round(llistaCantsDetallada.get(i).getFirst()/msAgrupacio));
          while (finalFragment - duradaFragment < 0) duradaFragment /= 2;
          iniciPujada = duradaFragment == 0;
          iniciFragment = finalFragment - duradaFragment;

          //Va tirant enrere a partir de l'inici del cant trobat per evitar haver-se'n
saltat la primera part a causa del pendent que tenen aquest tipus de cants (comencen
fluix i van augmentant la potència progressivament).
          for(int j = 0; j < 5; j++) {
            finalFragmentOriginal = finalFragment;
            while (!iniciPujada && duradaFragment != 0 && ((i == 0 && iniciFragment
>= 0) || (i > 0 && iniciFragment < (int) (Math.round(llistaCantsDetallada.get(i-
1).getSecond()/msAgrupacio))))) {
              iniciPujada =
amplitudsModificadesAbsolutesUltraReduides.get(iniciFragment) >=
amplitudsModificadesAbsolutesUltraReduides.get(finalFragment)*(1 -
(pendentPujada*(finalFragment-iniciFragment)/duradaFragmentOriginal)) ||
amplitudsModificadesAbsolutesUltraReduides.get(iniciFragment) <
mitjanesAmplitudsCants.get(i)*0.1;
              if (!iniciPujada) {
                if (finalFragment - duradaFragment > 0 && iniciFragment -
duradaFragment > 0) {
                  finalFragment -= duradaFragment;
                  iniciFragment -= duradaFragment;
                }
                else iniciPujada = true;
              }
            }
            //No ha aconseguit tirar més enrere. Per tant, ha de mirar si el cant
comença més tard.
            if (finalFragment == finalFragmentOriginal) {
              iniciFragment = finalFragmentOriginal;
              finalFragment = finalFragmentOriginal + duradaFragment*2;
            }

            if (iniciFragment * msAgrupacio >= llistaCantsDetallada.get(i).getFirst() ||
finalFragment * msAgrupacio > llistaCantsDetallada.get(i).getFirst() + msAgrupacio/2)
{
              break;
            }

            duradaFragment /= 2;

            iniciPujada = false;
          }

          //Si es tractava d'un cant tipus Hilaphura varipes o el curt que pot fer la
Cicadatra atra i ha pogut tirar l'inici més enrere, actualitza el cant.
          if (iniciFragment*msAgrupacio < llistaCantsDetallada.get(i).getFirst() &&
finalFragment*msAgrupacio <= llistaCantsDetallada.get(i).getFirst() + msAgrupacio/2)
{
            llistaCantsDetallada.set(i, new Pair(iniciFragment*msAgrupacio*1.0,
llistaCantsDetallada.get(i).getSecond()));
          }
        }
      }


      //Comprova si el cant ocupa tot el fitxer de so per evitar no tenir-lo en compte
si es dóna el cas.
      double mitjAmpl = 0.0;
      double minAmpl = Double.MAX_VALUE;
      double desvEstAmpl =
trobaDesviacioEstandardValors(amplitudsModificadesAbsolutesReduides025ms);

      for (Double amplitudsModificadesAbsolutesReduides25m :
amplitudsModificadesAbsolutesReduides025ms) {
        mitjAmpl += amplitudsModificadesAbsolutesReduides25m;
        if (amplitudsModificadesAbsolutesReduides25m < minAmpl) {
          minAmpl = amplitudsModificadesAbsolutesReduides25m;
        }
      }
      mitjAmpl /= amplitudsModificadesAbsolutesReduides025ms.size();

      double nivellConfiancaGlobalCants = Math.max(0, (1 -
trobaDesviacioEstandard(llistaCantsDetallada))) * Math.min(1, Math.max(0.2,
Math.log10(llistaCantsDetallada.size())));
      double maximaDurada = 0.0;

      for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
        if (llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst() >
maximaDurada) {
          maximaDurada = llistaCantsDetallada1.getSecond() -
llistaCantsDetallada1.getFirst();
        }
      }

      //El cant dura tot el fitxer de so.
      if (mitjAmpl > 4000 && minAmpl > 100 && desvEstAmpl < 0.5 &&
nivellConfiancaGlobalCants <= 0.2 && maximaDurada <= 250) {
        llistaCantsDetallada.clear();
        llistaCantsDetallada.add(new Pair(0.0,
amplitudsModificadesAbsolutesReduides025ms.size() / 4.0 - 1));
      }

      //Elimina els cants massa baixos (fa les comparacions amb els cants contigus i
quasi contigus).
      boolean avanca = false;
      double amplActual, amplPre, amplPre2, amplPost, amplPost2;
      //Primer cant
      while (!avanca && llistaCantsDetallada.size() > 1) {
        amplActual = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(0).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(0).getSecond() -
llistaCantsDetallada.get(0).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplPost = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(1).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(1).getSecond() -
llistaCantsDetallada.get(1).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);

        if (amplActual < amplPost * 0.2) {
          llistaCantsDetallada.remove(0);
        }
        else avanca = true;
      }
      avanca = false;
      //Segon cant
      while (!avanca && llistaCantsDetallada.size() > 2) {
        amplPre = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(0).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(0).getSecond() -
llistaCantsDetallada.get(0).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplActual = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(1).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(1).getSecond() -
llistaCantsDetallada.get(1).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplPost = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(2).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(2).getSecond() -
llistaCantsDetallada.get(2).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);

        if (amplActual < (amplPre*0.5 + amplPost*0.5) * 0.225 && amplActual <
amplPre * 0.425 && amplActual < amplPost * 0.425) {
          llistaCantsDetallada.remove(1);
        }
        else avanca = true;
      }
      //Cants interiors
      itAux = 2;
      while (itAux < llistaCantsDetallada.size() - 2) {
        amplPre2 = trobaMitjana((int) Math.round(llistaCantsDetallada.get(itAux-
2).getFirst()*4), (int) Math.round((llistaCantsDetallada.get(itAux-2).getSecond() -
llistaCantsDetallada.get(itAux-2).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplPre = trobaMitjana((int) Math.round(llistaCantsDetallada.get(itAux-
1).getFirst()*4), (int) Math.round((llistaCantsDetallada.get(itAux-1).getSecond() -
llistaCantsDetallada.get(itAux-1).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplActual = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(itAux).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(itAux).getSecond() -
llistaCantsDetallada.get(itAux).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplPost = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(itAux+1).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(itAux+1).getSecond() -
llistaCantsDetallada.get(itAux+1).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplPost2 = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(itAux+2).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(itAux+2).getSecond() -
```

```
llistaCantsDetallada.get(itAux+2).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);

        if (amplActual < (amplPre*0.4 + amplPost*0.4 + amplPre2*0.2 +
amplPost2*0.2) * 0.25 && amplActual < amplPre * 0.725 && amplActual < amplPost *
0.725 && amplActual < amplPre2 * 0.925 && amplActual < amplPost2 * 0.925) {
            llistaCantsDetallada.remove(itAux);
        }
        else itAux++;
      }
      //Penúltim cant
      if (itAux < llistaCantsDetallada.size()-1) {
        amplPre = trobaMitjana((int) Math.round(llistaCantsDetallada.get(itAux-
1).getFirst()*4), (int) Math.round((llistaCantsDetallada.get(itAux-1).getSecond() -
llistaCantsDetallada.get(itAux-1).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplActual = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(itAux).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(itAux).getSecond() -
llistaCantsDetallada.get(itAux).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
        amplPost = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(itAux+1).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(itAux+1).getSecond() -
llistaCantsDetallada.get(itAux+1).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);

        if (amplActual < (amplPre*0.5 + amplPost*0.5) * 0.225 && amplActual <
amplPre * 0.425 && amplActual < amplPost * 0.425) {
          llistaCantsDetallada.remove(itAux);
        }
        else itAux++;
        //Últim cant
        if (itAux < llistaCantsDetallada.size()) {
          amplPre = trobaMitjana((int) Math.round(llistaCantsDetallada.get(itAux-
1).getFirst()*4), (int) Math.round((llistaCantsDetallada.get(itAux-1).getSecond() -
llistaCantsDetallada.get(itAux-1).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);
          amplActual = trobaMitjana((int)
Math.round(llistaCantsDetallada.get(itAux).getFirst()*4), (int)
Math.round((llistaCantsDetallada.get(itAux).getSecond() -
llistaCantsDetallada.get(itAux).getFirst())*4),
amplitudsModificadesAbsolutesReduides025ms);

          if (amplActual < amplPre * 0.2) {
            llistaCantsDetallada.remove(itAux);
          }
        }
      }

      //0,25 milisegons
      pas = (int) Math.round(2*fs*0.00025);


      ArrayList<Double> particions = new ArrayList<>();

      int nParticions = 10;
      double minimaAmplitud = Double.MAX_VALUE;
      double maximaAmplitud = 0;

      ArrayList<Integer> classificacioAmplituds = new ArrayList<>();

      //Fiabilitat, NGP
      Pair<Double, Double> ngp;
      int millorFiabilitatNGP = -10;
      ArrayList<Pair<Pair<Integer, Integer>, Pair<Double, Double> > > llistaNGP = new
ArrayList<>();
      double coeficientNombreEnllacosIGrupsPolsos = 0.0;
      double definicioGrupsDePolsosAux = 0.0;

      if (!llistaCantsDetallada.isEmpty()) {
        if (llistaCantsDetallada.get(llistaCantsDetallada.size()-1).getSecond() > (int)
Math.round(4*(amplitudsModificadesAbsolutes.size())*1000.0/(fs*2) - 1)/4.0) {
          llistaCantsDetallada.set(llistaCantsDetallada.size()-1, new
Pair(llistaCantsDetallada.get(llistaCantsDetallada.size()-1).getFirst(), (int)
Math.round(4*(amplitudsModificadesAbsolutes.size())*1000.0/(fs*2) - 1)/4.0));
        }

        //Ordena els cants segons la seva durada, de llarg a curt.
        Pair<Integer, Double> maximaDuradaCantsContinusLyristes = new Pair(0,
0.0);

        for (Integer indexsCantsContinusPlebeju : indexsCantsContinusPlebejus) {
          if (llistaCantsDetallada.get(indexsCantsContinusPlebeju).getSecond() -
llistaCantsDetallada.get(indexsCantsContinusPlebeju).getFirst() >
maximaDuradaCantsContinusLyristes.getSecond()) {
            maximaDuradaCantsContinusLyristes = new
Pair(indexsCantsContinusPlebeju,
llistaCantsDetallada.get(indexsCantsContinusPlebeju).getSecond() -
llistaCantsDetallada.get(indexsCantsContinusPlebeju).getFirst());
          }
        }

        if (maximaDuradaCantsContinusLyristes.getSecond() > 500) {
          indexCantMesLlarg = maximaDuradaCantsContinusLyristes.getFirst();
        }
        else {
          indexCantMesLlarg = 0;

          for (int i = 0; i < llistaCantsDetallada.size(); i++) {
            if (llistaCantsDetallada.get(i).getSecond() -
llistaCantsDetallada.get(i).getFirst() >
llistaCantsDetallada.get(indexCantMesLlarg).getSecond() -
llistaCantsDetallada.get(indexCantMesLlarg).getFirst()) indexCantMesLlarg = i;
          }
        }


        if (llistaCantsDetallada.get(indexCantMesLlarg).getSecond() -
llistaCantsDetallada.get(indexCantMesLlarg).getFirst() > 50) {

          double mitjanaAux;
          for(int i = (int)
Math.round((llistaCantsDetallada.get(indexCantMesLlarg).getFirst()+10)*2.0*fs/1000)
; i < (llistaCantsDetallada.get(indexCantMesLlarg).getFirst()+220)*2.0*fs/1000 - pas;
i+=pas) {
            mitjanaAux = trobaMitjana(i, pas, amplitudsModificadesAbsolutes);
            if (mitjanaAux > maximaAmplitud) maximaAmplitud = mitjanaAux;
            if (mitjanaAux < minimaAmplitud) minimaAmplitud = mitjanaAux;
          }

          for(int i = 1; i <= nParticions; i++) {
            particions.add(minimaAmplitud + (maximaAmplitud -
minimaAmplitud)*i/nParticions);
            classificacioAmplituds.add(0);
          }

          int iterador;
          for(int i = (int)
Math.round((llistaCantsDetallada.get(indexCantMesLlarg).getFirst()+10)*2.0*fs/1000)
; i < (llistaCantsDetallada.get(indexCantMesLlarg).getFirst()+220)*2.0*fs/1000 - pas;
i+= pas) {
            iterador = 0;
            mitjanaAux = trobaMitjana(i, pas, amplitudsModificadesAbsolutes);
            while (iterador < nParticions-1 && mitjanaAux > particions.get(iterador))
{
              iterador++;
            }
            classificacioAmplituds.set(iterador,
classificacioAmplituds.get(iterador)+1);
          }
        }

        ArrayList<Pair<Double, Double> > llistaCantsDetalladaOrdenadaPerDurada =
new ArrayList<>();
        llistaCantsDetalladaOrdenadaPerDurada.addAll(llistaCantsDetallada);
        Collections.sort(llistaCantsDetalladaOrdenadaPerDurada, new
ComparadorParelles());

        //Durant un minut com a màxim, es dedica a cercar el NGP dels cants més
llargs que té.
        long momentAbansDeComencar = System.currentTimeMillis();
        long momentActual = momentAbansDeComencar;
        long duradaAnterior = 0;

        //No permet que hi hagi cants de més de 120 segons perquè trigaria massa a
trobar-ne el NGP.
        ArrayList<Pair<Double, Double> >
llistaCantsDetalladaOrdenadaPerDuradaCopia = new ArrayList<>();

llistaCantsDetalladaOrdenadaPerDuradaCopia.addAll(llistaCantsDetalladaOrdenadaPe
rDurada);

        for(int i = 0; i < llistaCantsDetalladaOrdenadaPerDuradaCopia.size(); i++) {
          if (llistaCantsDetalladaOrdenadaPerDuradaCopia.get(i).getSecond() -
llistaCantsDetalladaOrdenadaPerDuradaCopia.get(i).getFirst() > 120000) {
            llistaCantsDetalladaOrdenadaPerDuradaCopia.set(i, new
Pair(llistaCantsDetalladaOrdenadaPerDuradaCopia.get(i).getFirst(),
llistaCantsDetalladaOrdenadaPerDuradaCopia.get(i).getFirst()+120000.0));
          }
        }

        //amplitudsReduides025msCalculades serveix per saber si ja ha calculat les
amplitudsModificadesAbsolutesReduides o no.
        if (!amplitudsReduides025msCalculades) {
          for(int i = 0; i < amplitudsModificadesAbsolutes.size() - pas; i += pas) {

amplitudsModificadesAbsolutesReduides025ms.add(trobaMitjanaAmbValorsAbsoluts
(i, pas, amplitudsModificadesAbsolutes));
          }
        }
```

```
        int iterador = 0;
        double duradaTotalms = 0.0;
        double duradaParcialms;
        while (momentActual < momentAbansDeComencar + 60000 - duradaAnterior
&& iterador < llistaCantsDetalladaOrdenadaPerDurada.size() &&
llistaCantsDetalladaOrdenadaPerDurada.get(iterador).getSecond() -
llistaCantsDetalladaOrdenadaPerDurada.get(iterador).getFirst() > 250) {
            llistaNGP.add(trobaNGP(amplitudsModificades,
amplitudsModificadesAbsolutesReduides025ms, fs,
llistaCantsDetalladaOrdenadaPerDurada, iterador));
            if (llistaNGP.get(llistaNGP.size()-1).getSecond().getSecond() > 200 ||
llistaNGP.get(llistaNGP.size()-1).getSecond().getSecond() < 50)
              llistaNGP.remove(llistaNGP.size()-1);
            else {
              duradaParcialms =
llistaCantsDetalladaOrdenadaPerDurada.get(iterador).getSecond() -
llistaCantsDetalladaOrdenadaPerDurada.get(iterador).getFirst();
              duradaTotalms += duradaParcialms;
              definicioGrupsDePolsosAux += llistaNGP.get(llistaNGP.size()-
1).getSecond().getFirst() * duradaParcialms;
            }
            duradaAnterior = System.currentTimeMillis() - momentActual;
            momentActual = System.currentTimeMillis();
            iterador++;
        }


        //Filtra els NGP per evitar fer mitjanes amb valors incorrectes.
        //Pot ser que en cinc cants hagi sortit un valor corresponent a la Tibicina
garricola i en un hagi sortit corresponent a la Tibicina quadrisignata. En aquest cas no
hauria de fer mitjana, sinó quedar-se amb el valor que considera bo.
        int rangMinimCorsicaFairmairei = 56;
        int rangMinimHabitualCorsicaFairmairei = 59;
        int rangMaximHabitualCorsicaFairmairei = 65;
        int rangMaximCorsicaFairmairei = 68;

        int rangMinimGarricola = 63;
        int rangMinimHabitualGarricola = 66;
        int rangMaximHabitualGarricola = 71;
        int rangMaximGarricola = 74;

        int rangMinimHaematodes = 85;
        int rangMinimHabitualHaematodes = 90;
        int rangMaximHabitualHaematodes = 105;
        int rangMaximHaematodes = 110;

        int rangMinimQuadrisignata = 67;
        int rangMinimHabitualQuadrisignata = 73;
        int rangMaximHabitualQuadrisignata = 80;
        int rangMaximQuadrisignata = 85;

        int rangMinimTomentosa = 138;
        int rangMinimHabitualTomentosa = 145;
        int rangMaximHabitualTomentosa = 180;
        int rangMaximTomentosa = 185;

        int rangMinimBarbaraLusitanica = 95;
        int rangMinimHabitualBarbaraLusitanica = 110;
        int rangMaximHabitualBarbaraLusitanica = 200;
        int rangMaximBarbaraLusitanica = 215;

        double puntsCorsicaFairmairei = 0.0;
        double puntsGarricola = 0.0;
        double puntsHaematodes = 0.0;
        double puntsQuadrisignata = 0.0;
        double puntsTomentosa = 0.0;
        double puntsBarbaraLusitanica = 0.0;

        double ngpAux, puntsAux;
        for (Pair<Pair<Integer, Integer>, Pair<Double, Double>> llistaNGP1 : llistaNGP)
{
          ngpAux = llistaNGP1.getSecond().getSecond();
          if (llistaNGP1.getFirst().getFirst() == 0) {
            puntsAux = Math.min(1, Math.log10(llistaNGP1.getFirst().getSecond()));
          } else if (llistaNGP1.getFirst().getFirst() == -1) {
            puntsAux = Math.min(1, Math.log10(llistaNGP1.getFirst().getSecond())) *
0.2;
          } else {
            puntsAux = 0.0;
          }
          if (ngpAux >= rangMinimCorsicaFairmairei && ngpAux <=
rangMaximCorsicaFairmairei) {
            if (ngpAux >= rangMinimHabitualCorsicaFairmairei && ngpAux <=
rangMaximHabitualCorsicaFairmairei)
              puntsCorsicaFairmairei += puntsAux * 5;
            else
              puntsCorsicaFairmairei += puntsAux;
          }
          if (ngpAux >= rangMinimGarricola && ngpAux <= rangMaximGarricola) {
            if (ngpAux >= rangMinimHabitualGarricola && ngpAux <=
rangMaximHabitualGarricola)
              puntsGarricola += puntsAux * 5;
            else
              puntsGarricola += puntsAux;
          }
          if (ngpAux >= rangMinimHaematodes && ngpAux <=
rangMaximHaematodes) {
            if (ngpAux >= rangMinimHabitualHaematodes && ngpAux <=
rangMaximHabitualHaematodes)
              puntsHaematodes += puntsAux * 5;
            else
              puntsHaematodes += puntsAux;
          }
          if (ngpAux >= rangMinimQuadrisignata && ngpAux <=
rangMaximQuadrisignata) {
            if (ngpAux >= rangMinimHabitualQuadrisignata && ngpAux <=
rangMaximHabitualQuadrisignata)
              puntsQuadrisignata += puntsAux * 5;
            else
              puntsQuadrisignata += puntsAux;
          }
          if (ngpAux >= rangMinimTomentosa && ngpAux <= rangMaximTomentosa)
{
            if (ngpAux >= rangMinimHabitualTomentosa && ngpAux <=
rangMaximHabitualTomentosa)
              puntsTomentosa += puntsAux * 5;
            else
              puntsTomentosa += puntsAux;
          }
          if (ngpAux >= rangMinimBarbaraLusitanica && ngpAux <=
rangMaximBarbaraLusitanica) {
            if (ngpAux >= rangMinimHabitualBarbaraLusitanica && ngpAux <=
rangMaximHabitualBarbaraLusitanica)
              puntsBarbaraLusitanica += puntsAux * 1.25;
            else
              puntsBarbaraLusitanica += puntsAux * 0.25;
          }
        }

        double potencia = 0.75;
        puntsCorsicaFairmairei *= Math.pow(((mitjFreq < 8750) ? Math.max(0,
1.0*(mitjFreq - 7250) / (8750-7250)) : ((mitjFreq > 10500) ? Math.max(0, 1 -
1.0*(mitjFreq - 10500) / (12000-10500)) : 1)), potencia);
        puntsGarricola *= Math.pow(((mitjFreq < 7500) ? Math.max(0, 1.0*(mitjFreq
- 6000) / (7500-6000)) : ((mitjFreq > 9000) ? Math.max(0, 1 - 1.0*(mitjFreq - 9000) /
(10500-9000)) : 1)), potencia);
        puntsHaematodes *= Math.pow(((mitjFreq < 6000) ? Math.max(0,
1.0*(mitjFreq - 4500) / (6000-4500)) : ((mitjFreq > 8250) ? Math.max(0, 1 -
1.0*(mitjFreq - 8250) / (9750-8250)) : 1)), potencia);
        puntsQuadrisignata *= Math.pow(((mitjFreq < 8250) ? Math.max(0,
1.0*(mitjFreq - 6750) / (8250-6750)) : ((mitjFreq > 9500) ? Math.max(0, 1 -
1.0*(mitjFreq - 9500) / (11000-9500)) : 1)), potencia);
        puntsTomentosa *= Math.pow(((mitjFreq < 8250) ? Math.max(0,
1.0*(mitjFreq - 6750) / (8250-6750)) : ((mitjFreq > 9250) ? Math.max(0, 1 -
1.0*(mitjFreq - 9250) / (10750-9250)) : 1)), potencia);
        puntsBarbaraLusitanica *= Math.pow(((mitjFreq < 5500) ? Math.max(0,
1.0*(mitjFreq - 4000) / (5500-4000)) : ((mitjFreq > 7000) ? Math.max(0, 1 -
1.0*(mitjFreq - 7000) / (8500-7000)) : 1)), potencia);

        boolean seSalva;
        double puntsMaxims = Math.max(puntsCorsicaFairmairei,
Math.max(puntsGarricola, Math.max(puntsQuadrisignata,
Math.max(puntsTomentosa, puntsBarbaraLusitanica))));
        double minimNGPAccepto = 0;
        double maximNGPAccepto = Double.MAX_VALUE;

        if (puntsMaxims > 1) {
          if (puntsCorsicaFairmairei == puntsMaxims) {
            minimNGPAccepto = rangMinimHabitualCorsicaFairmairei / 1.15;
            maximNGPAccepto = rangMaximHabitualCorsicaFairmairei * 1.15;
          }
          else if (puntsGarricola == puntsMaxims) {
            minimNGPAccepto = rangMinimHabitualGarricola / 1.15;
            maximNGPAccepto = rangMaximHabitualGarricola * 1.15;
          }
          else if (puntsHaematodes == puntsMaxims) {
            minimNGPAccepto = rangMinimHabitualHaematodes / 1.15;
            maximNGPAccepto = rangMaximHabitualHaematodes * 1.15;
          }
          else if (puntsQuadrisignata == puntsMaxims) {
            minimNGPAccepto = rangMinimHabitualQuadrisignata / 1.15;
            maximNGPAccepto = rangMaximHabitualQuadrisignata * 1.15;
          }
          else if (puntsTomentosa == puntsMaxims) {
            minimNGPAccepto = rangMinimHabitualTomentosa / 1.15;
            maximNGPAccepto = rangMaximHabitualTomentosa * 1.15;
          }
          else if (puntsBarbaraLusitanica == puntsMaxims) {
            minimNGPAccepto = rangMinimHabitualBarbaraLusitanica / 1.15;
            maximNGPAccepto = rangMaximHabitualBarbaraLusitanica * 1.15;
          }
        }
```

```
        //Decideix amb quins valors de NGP es queda en cas que en tingui de
substancialment diferents.
        for(int i = 0; i < llistaNGP.size(); i++) {
          ngpAux = llistaNGP.get(i).getSecond().getSecond();
          seSalva = false;

          if (puntsMaxims < 1) {
            seSalva = true;
          }
          else if (ngpAux >= minimNGPAccepto && ngpAux <= maximNGPAccepto) {
            seSalva = true;
          }

          if (!seSalva) {
            llistaNGP.remove(i);
            i--;
          }
        }


        if (duradaTotalms > 0) definicioGrupsDePolsosAux /= duradaTotalms;

        //Recupera els cants que s'ha tallat.
        llistaCantsDetalladaOrdenadaPerDurada.clear();

llistaCantsDetalladaOrdenadaPerDurada.addAll(llistaCantsDetalladaOrdenadaPerDura
daCopia);

        //Agrupa els NGP per poder triar-ne el correcte.
        double maximNGP = 0;
        double minimNGP = Double.MAX_VALUE;
        int nParticionsNGP = 500;
        ArrayList<Double> particionsNGP = new ArrayList<>();
        ArrayList<Double> classificacioNGP = new ArrayList<>();
        for (Pair<Pair<Integer, Integer>, Pair<Double, Double>> llistaNGP1 : llistaNGP)
{
          if (llistaNGP1.getSecond().getSecond() > maximNGP) {
            maximNGP = llistaNGP1.getSecond().getSecond();
          }
          if (llistaNGP1.getSecond().getSecond() < minimNGP) {
            minimNGP = llistaNGP1.getSecond().getSecond();
          }
          if (llistaNGP1.getFirst().getFirst() > millorFiabilitatNGP) {
            millorFiabilitatNGP = llistaNGP1.getFirst().getFirst();
          }
        }

        for(int i = 1; i <= nParticionsNGP; i++) {
          particionsNGP.add(minimNGP + (maximNGP -
minimNGP)*i/nParticionsNGP);
          classificacioNGP.add(0.0);
        }

        for (Pair<Pair<Integer, Integer>, Pair<Double, Double>> llistaNGP1 : llistaNGP)
{
          if (llistaNGP1.getFirst().getFirst() == 0) {
            coeficientNombreEnllacosIGrupsPolsos +=
llistaNGP1.getFirst().getSecond();
          }
          else if (llistaNGP1.getFirst().getFirst() == -1) {
            coeficientNombreEnllacosIGrupsPolsos +=
llistaNGP1.getFirst().getSecond() * 0.2;
          }
        }

        if (coeficientNombreEnllacosIGrupsPolsos > 0)
          coeficientNombreEnllacosIGrupsPolsos = Math.min(1,
Math.log10(coeficientNombreEnllacosIGrupsPolsos) / 2.5);

        //No pot ponderar el mateix un NGP de fiabilitat -1 i 20 GDP+Enllaços que un
de fiabilitat 0 i 500 GDP+Enllaços.
        for (Pair<Pair<Integer, Integer>, Pair<Double, Double>> llistaNGP1 : llistaNGP)
{
          iterador = 0;
          while (iterador < nParticionsNGP-1 && llistaNGP1.getSecond().getSecond()
> particionsNGP.get(iterador)) {
            iterador++;
          }
          if (llistaNGP1.getFirst().getFirst() == 0) {
            //ln(GDP+Enllaços) per donar més fiabilitat als NGP obtinguts amb més
mostres.
            classificacioNGP.set(iterador, classificacioNGP.get(iterador) + 1.5 *
Math.log(llistaNGP1.getFirst().getSecond()));
          }
          else if (llistaNGP1.getFirst().getFirst() == -1) {
            classificacioNGP.set(iterador, classificacioNGP.get(iterador)+1.0);
          }
        }

        ArrayList<Double> llistaPerTrobarSD = new ArrayList<>();
        int nInsercions;
        for(int i = 0; i < classificacioNGP.size(); i++) {
          nInsercions = (int) Math.round(classificacioNGP.get(i));
          for(int j = 0; j < nInsercions; j++) {
            llistaPerTrobarSD.add(minimNGP + (maximNGP -
minimNGP)*(i+0.5)/nParticionsNGP);
          }
        }

        //Serà un indicador de la fiabilitat del NGP trobat.
        double desviacioEstandardNGP =
trobaDesviacioEstandardValors(llistaPerTrobarSD);

        double mitjanaNGP = 0;
        for (Double llistaPerTrobarSD1 : llistaPerTrobarSD) {
          mitjanaNGP += llistaPerTrobarSD1;
        }
        mitjanaNGP /= llistaPerTrobarSD.size();

        int nNGPs = 0;
        double NGPs = 0.0;
        for (Double llistaPerTrobarSD1 : llistaPerTrobarSD) {
          if (Math.abs(llistaPerTrobarSD1 - mitjanaNGP) / mitjanaNGP < 0.15) {
            NGPs += llistaPerTrobarSD1;
            nNGPs++;
          }
        }
        if (nNGPs > 0) NGPs /= nNGPs;

        ngp = new Pair(desviacioEstandardNGP, NGPs);
      }
      else {
        ngp = new Pair(0.0, 0.0);
      }


      int duradamsLimitCantCurtOLlarg = 700;


      //Declara totes les variables que tindrà en compte a l'hora de determinar
l'espècie de cigala.
      double nivellConfiancaSonCantsCurts;
      double nivellConfiancaSonCantsLlargs;
      double nivellConfiancaEsCantLlargIrregular;
      double duradaMitjanaCantCurt;
      double duradaMinimaCantCurt;
      double duradaMaximaCantCurt;
      double nivellConfiancaDuradaCantsCurts;
      double duradaMitjanaSilenci;
      double duradaMinimaSilenci;
      double duradaMaximaSilenci;
      double nivellConfiancaDuradaSilencis;
      double percentatgeCantsCurts;
      boolean hiHaCantsCurtsAmbCantLlargPrevi;
      double nivellConfiancaCantsCurtsAmbCantLlargPrevi;
      double duradaMitjanaCantCurtAmbCantLlargPrevi;
      double duradaMinimaCantCurtAmbCantLlargPrevi;
      double duradaMaximaCantCurtAmbCantLlargPrevi;
      double nivellConfiancaDuradaCantCurtAmbCantLlargPrevi;
      double duradaMitjanaCantLlargAmbCantCurtPosterior;
      double duradaMinimaCantLlargAmbCantCurtPosterior;
      double duradaMaximaCantLlargAmbCantCurtPosterior;
      double nivellConfiancaDuradaCantLlargAmbCantCurtPosterior;
      double duradaMitjanaSilenciEntreCantLlargICantCurtPosterior;
      double duradaMinimaSilenciEntreCantLlargICantCurtPosterior;
      double duradaMaximaSilenciEntreCantLlargICantCurtPosterior;
      double nivellConfiancaDuradaSilenciEntreCantLlargICantCurtPosterior;
      double pendentPrimeraMeitatCantsCurts;
      double pendentSegonaMeitatCantsCurts;
      double pendentPrimeraMeitatCantsLlargs;
      double pendentSegonaMeitatCantsLlargs;
      double correlacioPotenciaCantsILlargadaSilencis;
      double frequenciaMaximaMostreig;
      double frequenciaMitjana;
      int pic1Frequencia;
      int pic2Frequencia;
      int pic3Frequencia;
      double nombreGrupsDePolsosPerSegon;
      double nivellConfiancaNombreGrupsDePolsosPerSegon;
      double definicioGrupsDePolsos;
      double tempsDurada;
      double mitjanaAmplitud;
      double maximAmplitud;
      double mitjanaAmplitudCant;
      double mitjanaAmplitudSilencis;
      double proporcioSilenci;
      int nBatecsAlesOCantsIncorrectes;
      double nivellConfiancaCantsCO;
      double nivellConfiancaCantsLP;
```

```
        //Durades cants
        duradaMinimaCantCurt = Double.MAX_VALUE;
        duradaMaximaCantCurt = 0.0;
        duradaMitjanaCantCurt = 0.0;

        double duradaCantAux;
        int nCantsCurts = 0;
        int nCantsLlargs = 0;
        ArrayList<Double> llistaDuradesCantsCurts = new ArrayList<>();

        for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
          duradaCantAux = llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
          if (duradaCantAux < duradamsLimitCantCurtOLlarg) {
            nCantsCurts++;
            if (duradaCantAux < duradaMinimaCantCurt) duradaMinimaCantCurt = duradaCantAux;
            if (duradaCantAux > duradaMaximaCantCurt) duradaMaximaCantCurt = duradaCantAux;
            duradaMitjanaCantCurt += duradaCantAux;
            llistaDuradesCantsCurts.add(duradaCantAux);
          }
          else nCantsLlargs++;
        }

        if (nCantsCurts > 0) duradaMitjanaCantCurt /= nCantsCurts;

        if (nCantsCurts > 0)
          nivellConfiancaDuradaCantsCurts = Math.max(0, (1 - trobaDesviacioEstandardValors(llistaDuradesCantsCurts))) * Math.min(1, Math.max(0.2, Math.log10(nCantsCurts)));
        else
          nivellConfiancaDuradaCantsCurts = 0.0;

        if (nCantsLlargs == 0 && nCantsCurts == 0) percentatgeCantsCurts = 0.0;
        else percentatgeCantsCurts = 100.0 * nCantsCurts / (nCantsCurts + nCantsLlargs);


        //Cants curts amb cant llarg previ
        ArrayList<Pair<Double, Double> > cantsCurtsAmbCantLlargPrevi = new ArrayList<>();
        ArrayList<Pair<Double, Double> > cantsLlargsAmbCantCurtPosterior = new ArrayList<>();

        for(int i = 1; i < llistaCantsDetallada.size(); i++) {
          if (sonCantLlargICantCurt(llistaCantsDetallada, i)) {
            cantsCurtsAmbCantLlargPrevi.add(llistaCantsDetallada.get(i));
            cantsLlargsAmbCantCurtPosterior.add(llistaCantsDetallada.get(i-1));
          }
        }

        hiHaCantsCurtsAmbCantLlargPrevi = cantsCurtsAmbCantLlargPrevi.size() + cantsLlargsAmbCantCurtPosterior.size() >= nCantsCurts - 2;

        boolean sHaDeTornarAFer = true;

        duradaMinimaCantCurtAmbCantLlargPrevi = Double.MAX_VALUE;
        duradaMaximaCantCurtAmbCantLlargPrevi = 0.0;
        duradaMitjanaCantCurtAmbCantLlargPrevi = 0.0;

        duradaMinimaCantLlargAmbCantCurtPosterior = Double.MAX_VALUE;
        duradaMaximaCantLlargAmbCantCurtPosterior = 0.0;
        duradaMitjanaCantLlargAmbCantCurtPosterior = 0.0;

        duradaMinimaSilenci = Double.MAX_VALUE;
        duradaMaximaSilenci = 0.0;
        duradaMitjanaSilenci = 0.0;

        duradaMinimaSilenciEntreCantLlargICantCurtPosterior = Double.MAX_VALUE;
        duradaMaximaSilenciEntreCantLlargICantCurtPosterior = 0.0;
        duradaMitjanaSilenciEntreCantLlargICantCurtPosterior = 0.0;

        nivellConfiancaDuradaSilencis = 0.0;
        nivellConfiancaCantsCurtsAmbCantLlargPrevi = 0.0;

        //Treu els batecs d'ales o cants incorrectes.
        nBatecsAlesOCantsIncorrectes = 0;

        if (duradaMitjanaCantCurt > 50) {
          for(int i = 0; i < llistaCantsDetallada.size(); i++) {
            if (llistaCantsDetallada.get(i).getSecond() - llistaCantsDetallada.get(i).getFirst() < 12) {
              nBatecsAlesOCantsIncorrectes++;
              llistaCantsDetallada.remove(i);
              i--;
            }
          }
        }

        //Recalcula les durades mitjana i mínima i el nombre de cants curts.
        if (nBatecsAlesOCantsIncorrectes > 0) {
          duradaMitjanaCantCurt = 0.0;
          duradaMinimaCantCurt = Double.MAX_VALUE;
          nCantsCurts = 0;
          llistaDuradesCantsCurts.clear();
          for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
            duradaCantAux = llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
            if (duradaCantAux < duradamsLimitCantCurtOLlarg) {
              nCantsCurts++;
              if (duradaCantAux < duradaMinimaCantCurt) duradaMinimaCantCurt = duradaCantAux;
              duradaMitjanaCantCurt += duradaCantAux;
              llistaDuradesCantsCurts.add(duradaCantAux);
            }
          }
          if (nCantsCurts > 0) duradaMitjanaCantCurt /= nCantsCurts;
        }

        //Comprova les probabilitats que el cant analitzat tingui iteracions de cant llargues enllaçades cada una amb una de curta posterior.
        //Mentre hagi efectuat alguna correcció, continua al bucle, perquè la correcció en qüestió podria desencadenar algun altre canvi a la propera iteració.
        while (sHaDeTornarAFer) {

          if (!cantsCurtsAmbCantLlargPrevi.isEmpty() && hiHaCantsCurtsAmbCantLlargPrevi) {
            for (Pair<Double, Double> cantsCurtsAmbCantLlargPrevi1 : cantsCurtsAmbCantLlargPrevi) {
              duradaCantAux = cantsCurtsAmbCantLlargPrevi1.getSecond() - cantsCurtsAmbCantLlargPrevi1.getFirst();
              if (duradaCantAux < duradaMinimaCantCurtAmbCantLlargPrevi) duradaMinimaCantCurtAmbCantLlargPrevi = duradaCantAux;
              if (duradaCantAux > duradaMaximaCantCurtAmbCantLlargPrevi) duradaMaximaCantCurtAmbCantLlargPrevi = duradaCantAux;
              duradaMitjanaCantCurtAmbCantLlargPrevi += duradaCantAux;
            }

            duradaMitjanaCantCurtAmbCantLlargPrevi /= cantsCurtsAmbCantLlargPrevi.size();
          }

          if (!cantsCurtsAmbCantLlargPrevi.isEmpty())
            nivellConfiancaDuradaCantCurtAmbCantLlargPrevi = Math.max(0, (1 - trobaDesviacioEstandard(cantsCurtsAmbCantLlargPrevi))) * Math.min(1, Math.max(0.5, 1.25*Math.log10(cantsCurtsAmbCantLlargPrevi.size())) * Math.sqrt(1.0*cantsCurtsAmbCantLlargPrevi.size() / nCantsCurts));
          else
            nivellConfiancaDuradaCantCurtAmbCantLlargPrevi = 0.0;


          if (!cantsLlargsAmbCantCurtPosterior.isEmpty() && hiHaCantsCurtsAmbCantLlargPrevi) {
            for (Pair<Double, Double> cantsLlargsAmbCantCurtPosterior1 : cantsLlargsAmbCantCurtPosterior) {
              duradaCantAux = cantsLlargsAmbCantCurtPosterior1.getSecond() - cantsLlargsAmbCantCurtPosterior1.getFirst();
              if (duradaCantAux < duradaMinimaCantLlargAmbCantCurtPosterior) duradaMinimaCantLlargAmbCantCurtPosterior = duradaCantAux;
              if (duradaCantAux > duradaMaximaCantLlargAmbCantCurtPosterior) duradaMaximaCantLlargAmbCantCurtPosterior = duradaCantAux;
              duradaMitjanaCantLlargAmbCantCurtPosterior += duradaCantAux;
            }

            duradaMitjanaCantLlargAmbCantCurtPosterior /= cantsLlargsAmbCantCurtPosterior.size();
          }

          if (!cantsLlargsAmbCantCurtPosterior.isEmpty()) {
            ArrayList<Pair<Double, Double> > cantsMesOMenysLlargs = new ArrayList<>();
            for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
              if (llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst() > 320) {
                cantsMesOMenysLlargs.add(llistaCantsDetallada1);
              }
            }
            double nivellConfiancaCantsMesOMenysLlargs = Math.max(0, Math.sqrt(Math.max(0, 1 - trobaDesviacioEstandard(cantsMesOMenysLlargs))) * Math.min(1, Math.max(0.5, Math.log10(Math.max(1,cantsMesOMenysLlargs.size())))));
            nivellConfiancaDuradaCantLlargAmbCantCurtPosterior = Math.max(0, (1 - trobaDesviacioEstandard(cantsLlargsAmbCantCurtPosterior))) * Math.min(1, Math.max(0.5, Math.log10(cantsLlargsAmbCantCurtPosterior.size())) * Math.sqrt(nivellConfiancaCantsMesOMenysLlargs) * 1.0*cantsLlargsAmbCantCurtPosterior.size() / Math.max(1,cantsMesOMenysLlargs.size()));
          }
          else
            nivellConfiancaDuradaCantLlargAmbCantCurtPosterior = 0.0;
```

```
        //Durada silencis
        double duradaSilenciAux;
        int nSilencis = 0;
        int nSilencisEntreCantLlargICantCurtPosterior = 0;
        ArrayList<Double> llistaSilencis = new ArrayList<>();
        ArrayList<Double> llistaSilencisEntreCantLlargICantCurtPosterior = new ArrayList<>();

        for(int i = 1; i < llistaCantsDetallada.size(); i++) {
          if (hiHaCantsCurtsAmbCantLlargPrevi && sonCantLlargICantCurt(llistaCantsDetallada, i)) {
            duradaSilenciAux = llistaCantsDetallada.get(i).getFirst() - llistaCantsDetallada.get(i-1).getSecond();
            if (duradaSilenciAux < 1500) {
              nSilencisEntreCantLlargICantCurtPosterior++;
              if (duradaSilenciAux < duradaMinimaSilenciEntreCantLlargICantCurtPosterior) duradaMinimaSilenciEntreCantLlargICantCurtPosterior = duradaSilenciAux;
              if (duradaSilenciAux > duradaMaximaSilenciEntreCantLlargICantCurtPosterior) duradaMaximaSilenciEntreCantLlargICantCurtPosterior = duradaSilenciAux;
              duradaMitjanaSilenciEntreCantLlargICantCurtPosterior += duradaSilenciAux;
              llistaSilencisEntreCantLlargICantCurtPosterior.add(duradaSilenciAux);
            }
          }
          else {
            duradaSilenciAux = llistaCantsDetallada.get(i).getFirst() - llistaCantsDetallada.get(i-1).getSecond();
            if (duradaSilenciAux < 1500) {
              nSilencis++;
              if (duradaSilenciAux < duradaMinimaSilenci) duradaMinimaSilenci = duradaSilenciAux;
              if (duradaSilenciAux > duradaMaximaSilenci) duradaMaximaSilenci = duradaSilenciAux;
              duradaMitjanaSilenci += duradaSilenciAux;
              llistaSilencis.add(duradaSilenciAux);
            }
          }
        }

        if (nSilencis > 0) duradaMitjanaSilenci /= nSilencis;
        if (nSilencisEntreCantLlargICantCurtPosterior > 0) duradaMitjanaSilenciEntreCantLlargICantCurtPosterior /= nSilencisEntreCantLlargICantCurtPosterior;

        if (nSilencis > 0)
          nivellConfiancaDuradaSilencis = Math.max(0, (1 - trobaDesviacioEstandardValors(llistaSilencis))) * Math.min(1, Math.max(0.2, Math.log10(nSilencis)));
        else
          nivellConfiancaDuradaSilencis = 0.0;

        if (nSilencisEntreCantLlargICantCurtPosterior > 0)
          nivellConfiancaDuradaSilenciEntreCantLlargICantCurtPosterior = Math.max(0, (1 - trobaDesviacioEstandardValors(llistaSilencisEntreCantLlargICantCurtPosterior))) * Math.min(1, Math.max(0.5, Math.log10(nSilencisEntreCantLlargICantCurtPosterior)));
        else
          nivellConfiancaDuradaSilenciEntreCantLlargICantCurtPosterior = 0.0;


        //Confiança cants curts amb cant llarg previ
        if (nCantsCurts > 0)
          nivellConfiancaCantsCurtsAmbCantLlargPrevi = Math.min(1, (nivellConfiancaDuradaCantCurtAmbCantLlargPrevi*1.25 + nivellConfiancaDuradaCantLlargAmbCantCurtPosterior*0.75 + nivellConfiancaDuradaSilenciEntreCantLlargICantCurtPosterior*1.0)/3);
        else
          nivellConfiancaCantsCurtsAmbCantLlargPrevi = 0.0;

        boolean hiHaCantsCurtsAmbCantLlargPreviAnterior = hiHaCantsCurtsAmbCantLlargPrevi;

        hiHaCantsCurtsAmbCantLlargPrevi = nivellConfiancaCantsCurtsAmbCantLlargPrevi >= 0.15;

        sHaDeTornarAFer = !hiHaCantsCurtsAmbCantLlargPreviAnterior && hiHaCantsCurtsAmbCantLlargPrevi;
      }


      //Correcció de possibles cants "erronis":
      boolean hiHaHagutCanvis = true;
      while (nCantsCurts > 7 && !hiHaCantsCurtsAmbCantLlargPrevi && hiHaHagutCanvis) {
        ArrayList<Double> cantsCorrectes = new ArrayList<>();
        ArrayList<Pair<Double, Double> > cantsPressumptamentIncorrectes = new ArrayList<>();
        hiHaHagutCanvis = false;
        for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
          duradaCantAux = llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
          if (duradaCantAux < duradamsLimitCantCurtOLlarg) {
            if (duradaCantAux > duradaMitjanaCantCurt * 2.5 || duradaCantAux < duradaMitjanaCantCurt / 2.5) {
              cantsPressumptamentIncorrectes.add(llistaCantsDetallada1);
            } else {
              cantsCorrectes.add(duradaCantAux);
            }
          }
        }

        double nivellConfiancaDuradaCantsCurtsCorrectes = Math.max(0, (1 - trobaDesviacioEstandardValors(cantsCorrectes))) * Math.min(1, Math.max(0.2, Math.log10(cantsCorrectes.size())));

        if (!cantsPressumptamentIncorrectes.isEmpty() && nivellConfiancaDuradaCantsCurtsCorrectes > 0.3 && nivellConfiancaDuradaCantsCurtsCorrectes > nivellConfiancaDuradaCantsCurts) {
          duradaMinimaCantCurt = Double.MAX_VALUE;
          duradaMaximaCantCurt = 0;
          duradaMitjanaCantCurt = 0;
          for (Double cantsCorrecte : cantsCorrectes) {
            if (cantsCorrecte < duradaMinimaCantCurt) {
              duradaMinimaCantCurt = cantsCorrecte;
            }
            if (cantsCorrecte > duradaMaximaCantCurt) {
              duradaMaximaCantCurt = cantsCorrecte;
            }
            duradaMitjanaCantCurt += cantsCorrecte;
          }
          duradaMitjanaCantCurt /= cantsCorrectes.size();
          nivellConfiancaDuradaCantsCurts = nivellConfiancaDuradaCantsCurtsCorrectes;
          int iteradorAux = 0;
          for (Pair<Double, Double> cantsPressumptamentIncorrecte : cantsPressumptamentIncorrectes) {
            while (iteradorAux < llistaCantsDetallada.size() && llistaCantsDetallada.get(iteradorAux).getFirst() < cantsPressumptamentIncorrecte.getFirst()) {
              iteradorAux++;
            }
            if (iteradorAux < llistaCantsDetallada.size() && llistaCantsDetallada.get(iteradorAux).getFirst() == cantsPressumptamentIncorrecte.getFirst() && llistaCantsDetallada.get(iteradorAux).getSecond() == cantsPressumptamentIncorrecte.getSecond()) {
              llistaCantsDetallada.remove(iteradorAux);
              nCantsCurts--;
              hiHaHagutCanvis = true;
            }
          }
        }
      }

      correlacioPotenciaCantsILlargadaSilencis = calculaCorrelacioLlargadaCantsISilencis(llistaCantsDetallada, amplitudsModificadesAbsolutesReduides025ms);


      //Pendents cants
      ArrayList<ArrayList<Double> > pendentsCantsCurts = new ArrayList<>();
      ArrayList<ArrayList<Double> > pendentsCantsLlargs = new ArrayList< >();
      ArrayList<Double> pendentsAux;
      int nTalls = 8;
      double llargadaCant, milisegonTall, amplitudTallAnterior, amplitudTall, mitjaDuradaEnms;
      int durada;
      double factorConversio = fs*2/1000.0;

      for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
        if (llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst() > 120) {
          pendentsAux = new ArrayList<>();
          llargadaCant = llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
          mitjaDuradaEnms = llargadaCant/(nTalls*2.0);
          milisegonTall = llistaCantsDetallada1.getFirst() + llargadaCant * 1.0 / nTalls;
          durada = (int) Math.round(mitjaDuradaEnms*factorConversio);
          amplitudTallAnterior = trobaMitjana((int) Math.round(milisegonTall*factorConversio), durada/2, amplitudsModificadesAbsolutes);
          for (int j = 2; j < nTalls; j++) {
            milisegonTall = llistaCantsDetallada1.getFirst() + llargadaCant * j / nTalls;

            //Agafa dos milisegons per trobar-ne l'amplitud perquè les dades siguin més consistents. En cas que sigui l'últim, només agafa el milisegon anterior.
            durada = (int) Math.round(Math.min(mitjaDuradaEnms, llistaCantsDetallada1.getSecond() - milisegonTall + mitjaDuradaEnms/2) * factorConversio);
```

```
            amplitudTall = trobaMitjana((int) Math.round((milisegonTall-
mitjaDuradaEnms/2)*factorConversio), durada, amplitudsModificadesAbsolutes);
            pendentsAux.add(1.0*amplitudTall/amplitudTallAnterior);
            amplitudTallAnterior = amplitudTall;
          }
          if (llargadaCant < duradamsLimitCantCurtOLlarg)
            pendentsCantsCurts.add(pendentsAux);
          else
            pendentsCantsLlargs.add(pendentsAux);
        }
      }


      ArrayList<Double> mitjanesPendentsCurtsPrimeraMeitat = new ArrayList<>();
      ArrayList<Double> mitjanesPendentsCurtsSegonaMeitat = new ArrayList<>();
      ArrayList<Double> mitjanesPendentsLlargsPrimeraMeitat = new ArrayList<>();
      ArrayList<Double> mitjanesPendentsLlargsSegonaMeitat = new ArrayList<>();

      for (ArrayList<Double> pendentsCantsCurt : pendentsCantsCurts) {
        mitjanesPendentsCurtsPrimeraMeitat.add(trobaMitjana(0,
pendentsCantsCurt.size() / 2, pendentsCantsCurt));

mitjanesPendentsCurtsSegonaMeitat.add(trobaMitjana(pendentsCantsCurt.size() / 2
+ 1, pendentsCantsCurt.size() / 2, pendentsCantsCurt));
      }

      for (ArrayList<Double> pendentsCantsLlarg : pendentsCantsLlargs) {
        mitjanesPendentsLlargsPrimeraMeitat.add(trobaMitjana(0,
pendentsCantsLlarg.size() / 2, pendentsCantsLlarg));

mitjanesPendentsLlargsSegonaMeitat.add(trobaMitjana(pendentsCantsLlarg.size() / 2
+ 1, pendentsCantsLlarg.size() / 2, pendentsCantsLlarg));
      }

      pendentPrimeraMeitatCantsCurts =
trobaMitjana(0,mitjanesPendentsCurtsPrimeraMeitat.size(),
mitjanesPendentsCurtsPrimeraMeitat);
      pendentSegonaMeitatCantsCurts =
trobaMitjana(0,mitjanesPendentsCurtsSegonaMeitat.size(),
mitjanesPendentsCurtsSegonaMeitat);
      pendentPrimeraMeitatCantsLlargs =
trobaMitjana(0,mitjanesPendentsLlargsPrimeraMeitat.size(),
mitjanesPendentsLlargsPrimeraMeitat);
      pendentSegonaMeitatCantsLlargs =
trobaMitjana(0,mitjanesPendentsLlargsSegonaMeitat.size(),
mitjanesPendentsLlargsSegonaMeitat);


      //Tipus de cant (curt o llarg)
      nivellConfiancaSonCantsCurts = (percentatgeCantsCurts/100.0 +
((percentatgeCantsCurts == 100) ? 0.1 : 0.0) + nivellConfiancaDuradaCantsCurts / 2 +
nivellConfiancaDuradaSilencis / 5)/2;
      nivellConfiancaSonCantsLlargs = (1 - percentatgeCantsCurts/100.0 +
((percentatgeCantsCurts == 100) ? 0.0 : 0.1) - nivellConfiancaDuradaCantsCurts / 2 -
nivellConfiancaDuradaSilencis / 5)*2;

      if (nivellConfiancaSonCantsLlargs < 0) {
        nivellConfiancaSonCantsCurts -= nivellConfiancaSonCantsLlargs;
        nivellConfiancaSonCantsLlargs -= nivellConfiancaSonCantsLlargs;
      }
      if (nivellConfiancaSonCantsCurts < 0) {
        nivellConfiancaSonCantsCurts -= nivellConfiancaSonCantsCurts;
        nivellConfiancaSonCantsLlargs -= nivellConfiancaSonCantsCurts;
      }
      double sumaNivellsConfianca = nivellConfiancaSonCantsCurts +
nivellConfiancaSonCantsLlargs;

      nivellConfiancaSonCantsCurts /= sumaNivellsConfianca;
      nivellConfiancaSonCantsLlargs /= sumaNivellsConfianca;


      //Freqüències
      frequenciaMaximaMostreig = fsReal;
      frequenciaMitjana = mitjFreq;

      if (pics.size() == 3) {
        pic1Frequencia = pics.get(0);
        pic2Frequencia = pics.get(1);
        pic3Frequencia = pics.get(2);
      }
      else {
        pic1Frequencia = 0;
        pic3Frequencia = 0;
        if (pics.size() == 1) pic2Frequencia = pics.get(0);
        else pic2Frequencia = 0;
      }


      //NGP
      nombreGrupsDePolsosPerSegon = ngp.getSecond();
      if (millorFiabilitatNGP == 0) {
        nivellConfiancaNombreGrupsDePolsosPerSegon =
coeficientNombreEnllacosIGrupsPolsos * Math.max(0, (1 - ngp.getFirst()));
      }
      else if (millorFiabilitatNGP == -1) {
        nivellConfiancaNombreGrupsDePolsosPerSegon =
coeficientNombreEnllacosIGrupsPolsos * Math.max(0, (1 - ngp.getFirst())) * 0.5;
      }
      else {
        nivellConfiancaNombreGrupsDePolsosPerSegon = 0.0;
      }

      definicioGrupsDePolsos = Math.min(1, 50.0*definicioGrupsDePolsosAux);

      mitjanaAmplitud = 0.0;
      maximAmplitud = 0.0;
      for(int i = 0; i < amplitudsModificadesAbsolutesReduides.size(); i++) {
        mitjanaAmplitud += amplitudsModificadesAbsolutesReduides.get(i);
        if (amplitudsModificadesAbsolutesReduides.get(i) > maximAmplitud)
maximAmplitud = amplitudsModificadesAbsolutesReduides.get(i);
      }

      mitjanaAmplitud /= amplitudsModificadesAbsolutesReduides.size();



      //Amplituds i durades
      mitjanaAmplitudCant = 0.0;
      mitjanaAmplitudSilencis = 0.0;
      duradaTotalCants = 0.0;
      double duradaTotalSilencis = 0.0;


      if (!llistaCantsDetallada.isEmpty()) {
        //Silenci abans del primer cant
        double duradaAuxSilenci = llistaCantsDetallada.get(0).getFirst();
        mitjanaAmplitudSilencis += trobaMitjana(0, (int)
Math.round(duradaAuxSilenci*4), amplitudsModificadesAbsolutesReduides025ms) *
duradaAuxSilenci;
        duradaTotalSilencis += duradaAuxSilenci;

        double duradaAuxCant = llistaCantsDetallada.get(0).getSecond() -
llistaCantsDetallada.get(0).getFirst();
        mitjanaAmplitudCant += trobaMitjana((int)
Math.round(llistaCantsDetallada.get(0).getFirst()*4), (int)
Math.round(duradaAuxCant*4), amplitudsModificadesAbsolutesReduides025ms) *
duradaAuxCant;
        duradaTotalCants += duradaAuxCant;

        for(int i = 1; i < llistaCantsDetallada.size(); i++) {
          duradaAuxSilenci = llistaCantsDetallada.get(i).getFirst() -
llistaCantsDetallada.get(i-1).getSecond();
          mitjanaAmplitudSilencis += trobaMitjana((int)
Math.round(llistaCantsDetallada.get(i-1).getSecond()*4), (int)
Math.round(duradaAuxSilenci*4), amplitudsModificadesAbsolutesReduides025ms) *
duradaAuxSilenci;
          duradaTotalSilencis += duradaAuxSilenci;

          duradaAuxCant = llistaCantsDetallada.get(i).getSecond() -
llistaCantsDetallada.get(i).getFirst();
          mitjanaAmplitudCant += trobaMitjana((int)
Math.round(llistaCantsDetallada.get(i).getFirst()*4), (int)
Math.round(duradaAuxCant*4), amplitudsModificadesAbsolutesReduides025ms) *
duradaAuxCant;
          duradaTotalCants += duradaAuxCant;
        }

        //Silenci després de l'últim cant
        duradaAuxSilenci = Math.max(0,
amplitudsModificadesAbsolutesReduides025ms.size()/4 -
llistaCantsDetallada.get(llistaCantsDetallada.size()-1).getSecond());
        mitjanaAmplitudSilencis += trobaMitjana((int)
Math.round(llistaCantsDetallada.get(llistaCantsDetallada.size()-1).getSecond()*4),
(int) Math.round(duradaAuxSilenci*4),
amplitudsModificadesAbsolutesReduides025ms) * duradaAuxSilenci;
        duradaTotalSilencis += duradaAuxSilenci;

        if (duradaTotalCants >= 0)
          mitjanaAmplitudCant /= duradaTotalCants;
        if (duradaTotalSilencis >= 0)
          mitjanaAmplitudSilencis /= duradaTotalSilencis;

        proporcioSilenci = duradaTotalSilencis / (duradaTotalSilencis +
duradaTotalCants);
      }
      else {
        proporcioSilenci = 1.0;
      }
```

```
        //Assegura que no es tracti d'un cant llarg d'amplitud irregular per evitar considerar-lo com a diversos cants curts.
        ArrayList<Double> llistaDuradesAux = new ArrayList<>();
        ArrayList<Double> llistaDuradesCOAux = new ArrayList<>();
        ArrayList<Double> llistaDuradesLPAux = new ArrayList<>();
        for (Pair<Double, Double> llistaCantsDetallada1 : llistaCantsDetallada) {
            duradaCantAux = llistaCantsDetallada1.getSecond() - llistaCantsDetallada1.getFirst();
            if (duradaCantAux < duradamsLimitCantCurtOLlarg) {
                llistaDuradesAux.add(duradaCantAux);
                if (duradaCantAux >= 50 && duradaCantAux <= 170) llistaDuradesCOAux.add(duradaCantAux);
                if (duradaCantAux >= 35 && duradaCantAux <= 80) llistaDuradesLPAux.add(duradaCantAux);
            }
        }
        nivellConfiancaEsCantLlargIrregular = (Math.max(0, Math.min(1, Math.log10(nBatecsAlesOCantsIncorrectes)))*0.75 + Math.max(0, Math.min(1, 1+(2-mitjanaAmplitudCant/mitjanaAmplitudSilencis)/3))*1.25 + Math.max(0, Math.min(1, trobaDesviacioEstandardValors(llistaDuradesAux))) + nivellConfiancaSonCantsLlargs*0.5)/3.5;
        if (nivellConfiancaEsCantLlargIrregular > 0.65) {
        }

        //Comprova si les característiques dels cants curts encaixen amb els d'una Cicada orni.
        if (!llistaDuradesCOAux.isEmpty())
            nivellConfiancaCantsCO = trobaDesviacioEstandardValors(llistaDuradesCOAux) * Math.max(0, Math.min(1, Math.log(llistaDuradesCOAux.size())));
        else nivellConfiancaCantsCO = 0.0;

        //Comprova si les característiques dels cants curts encaixen amb els d'una Lyristes plebejus.
        if (!llistaDuradesLPAux.isEmpty())
            nivellConfiancaCantsLP = trobaDesviacioEstandardValors(llistaDuradesLPAux) * Math.max(0, Math.min(1, Math.log(llistaDuradesLPAux.size())));
        else nivellConfiancaCantsLP = 0.0;

        //Durada total de l'enregistrament
        tempsDurada = duradaEnSegons;

        //Troba els resultats de semblança amb cada espècie a partir dels valors obtinguts per a cada paràmetre del cant.
        resultatsIdentificacio = trobaResultatsIdentificacio(resultatsIdentificacio,
                                fitxer,
                                nivellConfiancaSonCantsCurts,
                                nivellConfiancaSonCantsLlargs,
                                nivellConfiancaEsCantLlargIrregular,
                                duradaMitjanaCantCurt,
                                duradaMitjanaSilenci,
                                percentatgeCantsCurts,
                                nivellConfiancaCantsCurtsAmbCantLlargPrevi,
                                duradaMitjanaCantCurtAmbCantLlargPrevi,
                                duradaMitjanaCantLlargAmbCantCurtPosterior,
                                duradaMitjanaSilenciEntreCantLlargICantCurtPosterior,
                                pendentPrimeraMeitatCantsCurts,
                                pendentSegonaMeitatCantsCurts,
                                pendentPrimeraMeitatCantsLlargs,
                                pendentSegonaMeitatCantsLlargs,
                                correlacioPotenciaCantsILlargadaSilencis,
                                frequenciaMaximaMostreig,
                                frequenciaMitjana,
                                pic1Frequencia,
                                pic2Frequencia,
                                pic3Frequencia,
                                nombreGrupsDePolsosPerSegon,
                                nivellConfiancaNombreGrupsDePolsosPerSegon,
                                definicioGrupsDePolsos,
                                tempsDurada,
                                proporcioSilenci,
                                nBatecsAlesOCantsIncorrectes,
                                llistaCantsDetallada,
                                nivellConfiancaCantsCO,
                                nivellConfiancaCantsLP,
                                nivellConfiancaDuradaCantsCurts,
                                nivellConfiancaDuradaSilencis,
                                mitjanaAmplitud,
                                mitjanaAmplitudCant);
    }
    catch(IOException ex) {
        resultatsIdentificacio += "Problema al fitxer " + fitxer.getName() + ": " + ex.getClass().toString() + ": " + ex.getMessage() + ", línia " + Thread.currentThread().getStackTrace()[2].getLineNumber() + "\n\n";
    }

    return resultatsIdentificacio;
  }

  /**
    @pre ---
    @post Retorna els resultats d'identificació corresponents al fitxer fitxer a partir de la resta de paràmetres entrats.
    @return Els resultats d'identificació corresponents al fitxer fitxer a partir de la resta de paràmetres entrats
    @param resultatsIdentificacio String que conté el nom del fitxer en qüestió per afegir-hi posteriorment els resultats obtinguts
    @param fitxer Fitxer que s'analitza
    @param nivellConfiancaSonCantsCurts Nivell de confiança que els cants siguin curts
    @param nivellConfiancaSonCantsLlargs Nivell de confiança que els cants siguin llargs
    @param nivellConfiancaEsCantLlargIrregular Nivell de confiança que cants siguin llargs i amb freqüències irregulars
    @param duradaMitjanaCantCurt Durada mitjana dels cants curts
    @param duradaMitjanaSilenci Durada mitjana dels silencis entre els cants
    @param percentatgeCantsCurts Percentatge de cants curts sobre el total de cants
    @param nivellConfiancaCantsCurtsAmbCantLlargPrevi Nivell de confiança que els cants són llargs amb cants curts enllaçats
    @param duradaMitjanaCantCurtAmbCantLlargPrevi Durada mitjana dels cants curts enllaçats a un cant llarg
    @param duradaMitjanaCantLlargAmbCantCurtPosterior Durada mitjana dels cants llargs amb un cant curt enllaçat
    @param duradaMitjanaSilenciEntreCantLlargICantCurtPosterior Durada mitjana dels silencis entre els cants llarg i curt enllaçats
    @param pendentPrimeraMeitatCantsCurts Pendent de l'amplitud de la primera meitat dels cants curts
    @param pendentSegonaMeitatCantsCurts Pendent de l'amplitud de la segon meitat dels cants curts
    @param pendentPrimeraMeitatCantsLlargs Pendent de l'amplitud de la primera meitat dels cants llargs
    @param pendentSegonaMeitatCantsLlargs Pendent de l'amplitud de la segona meitat dels cants llargs
    @param correlacioPotenciaCantsILlargadaSilencis Correlació entre la potència dels cants i la llargada dels silencis contigus
    @param frequenciaMaximaMostreig Freqüència de mostreig real
    @param frequenciaMitjana Freqüència mitjana del acnt
    @param pic1Frequencia Primer pic de freqüència del cant
    @param pic2Frequencia Segon pic de freqüència del cant
    @param pic3Frequencia Tercer pic de freqüència del cant
    @param nombreGrupsDePolsosPerSegon Nombre de grups de polsos per segon als cants llargs
    @param nivellConfiancaNombreGrupsDePolsosPerSegon Nivell confiança en el nombre de grups de polsos per segon als cants llargs
    @param definicioGrupsDePolsos Definició dels grups de polsos (com més definició, més fàcilment distingibles són)
    @param tempsDurada Durada de la gravació
    @param proporcioSilenci Proporció de temps de silenci respecte al temps total de la gravació
    @param nBatecsAlesOCantsIncorrectes Nombre de batecs d'ales o curts increments sobtats d'amplitud interpretats com a cant
    @param llistaCantsDetallada Llista que conté els moments d'inici i de final de cada iteracio de cant de l'enregistrament
    @param nivellConfiancaCantsCO Nivell de similitud entre els cants curts analitzats i els d'una Cicada orni
    @param nivellConfiancaCantsLP Nivell de similitud entre els cants curts analitzats i els d'una Lyristes plebejus
    @param nivellConfiancaDuradaCantsCurts Nivell de confiança en la durada dels cants curts
    @param nivellConfiancaDuradaSilencis Nivell de confiança en la durada dels silencis
    @param mitjanaAmplitud Amplitud mitjana del fitxer de so
    @param mitjanaAmplitudCant Amplitud mitjana del cant continguts al fitxer de so
   */
   private static String trobaResultatsIdentificacio(String resultatsIdentificacio,
      File fitxer,
      double nivellConfiancaSonCantsCurts,
      double nivellConfiancaSonCantsLlargs,
      double nivellConfiancaEsCantLlargIrregular,
      double duradaMitjanaCantCurt,
      double duradaMitjanaSilenci,
      double percentatgeCantsCurts,
      double nivellConfiancaCantsCurtsAmbCantLlargPrevi,
      double duradaMitjanaCantCurtAmbCantLlargPrevi,
      double duradaMitjanaCantLlargAmbCantCurtPosterior,
      double duradaMitjanaSilenciEntreCantLlargICantCurtPosterior,
      double pendentPrimeraMeitatCantsCurts,
      double pendentSegonaMeitatCantsCurts,
      double pendentPrimeraMeitatCantsLlargs,
      double pendentSegonaMeitatCantsLlargs,
      double correlacioPotenciaCantsILlargadaSilencis,
      double frequenciaMaximaMostreig,
      double frequenciaMitjana,
      int pic1Frequencia,
      int pic2Frequencia,
      int pic3Frequencia,
      double nombreGrupsDePolsosPerSegon,
      double nivellConfiancaNombreGrupsDePolsosPerSegon,
      double definicioGrupsDePolsos,
      double tempsDurada,
      double proporcioSilenci,
      int nBatecsAlesOCantsIncorrectes,
```

```
    ArrayList<Pair<Double, Double> > llistaCantsDetallada,
    double nivellConfiancaCantsCO,
    double nivellConfiancaCantsLP,
    double nivellConfiancaDuradaCantsCurts,
    double nivellConfiancaDuradaSilencis,
    double mitjanaAmplitud,
    double mitjanaAmplitudCant) {


    //Calcula les semblances amb les diferents espècies de cigala:
    ArrayList<Pair<Double, Double> > llistaResultatsSemblancesCigales = new
ArrayList<>();

    ArrayList<String> especies = new ArrayList<>();
    especies.add("Cicada orni");
    especies.add("Cicada barbara lusitanica");
    especies.add("Cicadatra atra");
    especies.add("Cicadetta brevipennis");
    especies.add("Cicadetta cerdaniensis");
    especies.add("Hilaphura varipes");
    especies.add("Lyristes plebejus");
    especies.add("Tettigettalna argentata");
    especies.add("Tettigettula pygmea");
    especies.add("Tibicina corsica fairmairei");
    especies.add("Tibicina garricola");
    especies.add("Tibicina haematodes");
    especies.add("Tibicina quadrisignata");
    especies.add("Tibicina tomentosa");

    for (String especie : especies) {
      llistaResultatsSemblancesCigales.add(trobaSemblancaAmbCigala(
          especie,
          nivellConfiancaSonCantsCurts,
          nivellConfiancaSonCantsLlargs,
          nivellConfiancaEsCantLlargIrregular,
          duradaMitjanaCantCurt,
          duradaMitjanaSilenci,
          percentatgeCantsCurts,
          nivellConfiancaCantsCurtsAmbCantLlargPrevi,
          duradaMitjanaCantCurtAmbCantLlargPrevi,
          duradaMitjanaCantLlargAmbCantCurtPosterior,
          duradaMitjanaSilenciEntreCantLlargICantCurtPosterior,
          pendentPrimeraMeitatCantsCurts,
          pendentSegonaMeitatCantsCurts,
          pendentPrimeraMeitatCantsLlargs,
          pendentSegonaMeitatCantsLlargs,
          correlacioPotenciaCantsILlargadaSilencis,
          frequenciaMaximaMostreig,
          frequenciaMitjana,
          pic1Frequencia,
          pic2Frequencia,
          pic3Frequencia,
          nombreGrupsDePolsosPerSegon,
          nivellConfiancaNombreGrupsDePolsosPerSegon,
          definicioGrupsDePolsos,
          tempsDurada,
          proporcioSilenci,
          nBatecsAlesOCantsIncorrectes,
          llistaCantsDetallada.size(),
          mitjanaAmplitud,
          mitjanaAmplitudCant));
    }

    Pair<Double, Double> resultatSemblancaCicadaOrni =
llistaResultatsSemblancesCigales.get(0);
    Pair<Double, Double> resultatSemblancaCicadaBarbaraLusitanica =
llistaResultatsSemblancesCigales.get(1);
    Pair<Double, Double> resultatSemblancaCicadatraAtra =
llistaResultatsSemblancesCigales.get(2);
    Pair<Double, Double> resultatSemblancaCicadettaBrevipennis =
llistaResultatsSemblancesCigales.get(3);
    Pair<Double, Double> resultatSemblancaCicadettaCerdaniensis =
llistaResultatsSemblancesCigales.get(4);
    Pair<Double, Double> resultatSemblancaHilaphuraVaripes =
llistaResultatsSemblancesCigales.get(5);
    Pair<Double, Double> resultatSemblancaLyristesPlebejus =
llistaResultatsSemblancesCigales.get(6);
    Pair<Double, Double> resultatSemblancaTettigettalnaArgentata =
llistaResultatsSemblancesCigales.get(7);
    Pair<Double, Double> resultatSemblancaTettigettulaPygmea =
llistaResultatsSemblancesCigales.get(8);
    Pair<Double, Double> resultatSemblancaTibicinaCorsicaFairmairei =
llistaResultatsSemblancesCigales.get(9);
    Pair<Double, Double> resultatSemblancaTibicinaGarricola =
llistaResultatsSemblancesCigales.get(10);
    Pair<Double, Double> resultatSemblancaTibicinaHaematodes =
llistaResultatsSemblancesCigales.get(11);
    Pair<Double, Double> resultatSemblancaTibicinaQuadrisignata =
llistaResultatsSemblancesCigales.get(12);
    Pair<Double, Double> resultatSemblancaTibicinaTomentosa =
llistaResultatsSemblancesCigales.get(13);

    //En cas que hi hagi dubtes entre dues espècies concretes, mira de resoldre'ls:
    double semblCO = resultatSemblancaCicadaOrni.getFirst();
    double semblCB = resultatSemblancaCicadaBarbaraLusitanica.getFirst();
    double semblCA = resultatSemblancaCicadatraAtra.getFirst();
    double semblCBr = resultatSemblancaCicadettaBrevipennis.getFirst();
    double semblCC = resultatSemblancaCicadettaCerdaniensis.getFirst();
    double semblHV = resultatSemblancaHilaphuraVaripes.getFirst();
    double semblLP = resultatSemblancaLyristesPlebejus.getFirst();
    double semblTA = resultatSemblancaTettigettalnaArgentata.getFirst();
    double semblTP = resultatSemblancaTettigettulaPygmea.getFirst();
    double semblTC = resultatSemblancaTibicinaCorsicaFairmairei.getFirst();
    double semblTG = resultatSemblancaTibicinaGarricola.getFirst();
    double semblTH = resultatSemblancaTibicinaHaematodes.getFirst();
    double semblTQ = resultatSemblancaTibicinaQuadrisignata.getFirst();
    double semblTT = resultatSemblancaTibicinaTomentosa.getFirst();

    //Increment de freqüència per a totes les espècies per a les quals fa comparacions
per ajudar-se en la identificació.
    //Augmenta la freqüència mitjana en cas que la freqüència de mostreig real sigui
massa baixa per haver agafat tot l'espectre freqüencial de l'espècie, amb la qual cosa
la mitjana surt més baixa i s'ha d'incrementar manualment.
    double incrementFreqCO = 1600*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-7000)/17000),5);
    double incrementFreqCB = 1600*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),5);
    double incrementFreqCA = 1500*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),10);
    double incrementFreqCC = 8000*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),2.7);
    double incrementFreqHV = 2200*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),5);
    double incrementFreqLP = 3000*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-7000)/17000),5);
    double incrementFreqTA = 3700*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-11000)/13000),4);
    double incrementFreqTP = 4500*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-16000)/8000),4);
    double incrementFreqTC = 1500*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),10);
    double incrementFreqTG = 1200*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),6);
    double incrementFreqTQ = 1500*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),8);
    double incrementFreqTT = 1200*Math.pow((1-
(Math.min(frequenciaMaximaMostreig,24000)-9000)/15000),8);

    //Comparacions entre diferents espècies per determinar millor les semblances
amb cada una en cas que no estigui clar

    //Cicada orni / Cicada barbara lusitanica
    if (semblCO > 0.25 && semblCB > 0.25) {
      double semblOriginalCO = semblCO;
      double semblOriginalCB = semblCB;

      double coefSonCantsLlargs = (nivellConfiancaSonCantsLlargs * 2 - 1) * 0.5;
      if (coefSonCantsLlargs > 0) {
        semblCO /= 1 + coefSonCantsLlargs;
        semblCB *= 1 + coefSonCantsLlargs;
      }
      else {
        semblCO *= 1 - coefSonCantsLlargs;
        semblCB /= 1 - coefSonCantsLlargs;
      }

      if (frequenciaMaximaMostreig >= 9000) {
        double coefFreq = ((frequenciaMitjana + incrementFreqCO - 6375) / 6375) *
0.2;
        if (coefFreq > 0) {
          semblCO /= 1 + coefFreq;
          semblCB *= 1 + coefFreq;
        }
        else {
          semblCO *= 1 - coefFreq;
          semblCB /= 1 - coefFreq;
        }
        coefFreq = ((frequenciaMitjana + incrementFreqCB - 6375) / 6375) * 0.2;
        if (coefFreq > 0) {
          semblCO /= 1 + coefFreq;
          semblCB *= 1 + coefFreq;
        }
        else {
          semblCO *= 1 - coefFreq;
          semblCB /= 1 - coefFreq;
        }
      }

      double coefPropSil = ((proporcioSilenci - 0.25) / 0.25) * 0.5;
      if (coefPropSil > 0) {
        semblCO *= 1 + coefPropSil;
        semblCB /= 1 + coefPropSil;
```

```
        }
        else {
            semblCO /= 1 - coefPropSil;
            semblCB *= 1 - coefPropSil;
        }

        double coefCantsCO = (nivellConfiancaCantsCO * 2 - 1) * 1.0;
        if (coefCantsCO > 0) {
            semblCO *= 1 + coefCantsCO;
            semblCB /= 1 + coefCantsCO;
        }
        else {
            semblCO /= 1 - coefCantsCO;
            semblCB *= 1 - coefCantsCO;
        }

        semblCO = Math.max(0, Math.min(1, semblCO));
        semblCB = Math.max(0, Math.min(1, semblCB));

        if (semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 ||
semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblCO > semblOriginalCO) {
                semblCB *= semblOriginalCO;
                semblCB /= semblCO;
                semblCO = semblOriginalCO;
            }
            else if (semblCB > semblOriginalCB) {
                semblCO *= semblOriginalCB;
                semblCO /= semblCB;
                semblCB = semblOriginalCB;
            }
        }

        resultatSemblancaCicadaOrni = new Pair(semblCO,
resultatSemblancaCicadaOrni.getSecond());
        resultatSemblancaCicadaBarbaraLusitanica = new Pair(semblCB,
resultatSemblancaCicadaBarbaraLusitanica.getSecond());
    }

    //Cicada orni / Cicadatra atra
    if (semblCO > 0.25 && semblCA > 0.25) {
        double semblOriginalCO = semblCO;
        double semblOriginalCA = semblCA;

        if (duradaMitjanaCantCurt > 0) {
            double coefDuradaMitjanaCant = ((170 - duradaMitjanaCantCurt) / 170) * 1.3
* nivellConfiancaDuradaCantsCurts;
            if (coefDuradaMitjanaCant > 0) {
                semblCO *= 1 + coefDuradaMitjanaCant;
                semblCA /= 1 + coefDuradaMitjanaCant;
            }

        }
        if (duradaMitjanaSilenci > 0) {
            double coefDuradaMitjanaSilenci = Math.max(0, ((120 -
duradaMitjanaSilenci) / 120) * 0.9) * nivellConfiancaDuradaSilencis;
            if (coefDuradaMitjanaSilenci > 0) {
                semblCO *= 1 + coefDuradaMitjanaSilenci;
                semblCA /= 1 + coefDuradaMitjanaSilenci;
            }
            else {
                coefDuradaMitjanaSilenci = Math.max(0, ((duradaMitjanaSilenci - 200) /
200) * 0.1) * nivellConfiancaDuradaSilencis;
                if (coefDuradaMitjanaSilenci > 0) {
                    semblCO /= 1 + coefDuradaMitjanaSilenci;
                    semblCA *= 1 + coefDuradaMitjanaSilenci;
                }
            }
        }
        double coefProporcioSilenci = Math.max(0, ((proporcioSilenci - 0.5)) * 0.4);
        if (coefProporcioSilenci > 0) {
            semblCO *= 1 + coefProporcioSilenci;
            semblCA /= 1 + coefProporcioSilenci;
        }

        if (semblCO > 1) {
            double varAux = semblCO;
            semblCO /= varAux;
            semblCA /= varAux;
        }
        if (semblCA > 1) {
            double varAux = semblCA;
            semblCO /= varAux;
            semblCA /= varAux;
        }

        semblCO = Math.max(0, Math.min(1, semblCO));
        semblCA = Math.max(0, Math.min(1, semblCA));

        if (semblCB > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 ||
semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblCO > semblOriginalCO) {
                semblCA *= semblOriginalCO;
                semblCA /= semblCO;
                semblCO = semblOriginalCO;
            }
            else if (semblCA > semblOriginalCA) {
                semblCO *= semblOriginalCA;
                semblCO /= semblCA;
                semblCA = semblOriginalCA;
            }
        }

        resultatSemblancaCicadaOrni = new Pair(semblCO,
resultatSemblancaCicadaOrni.getSecond());
        resultatSemblancaCicadatraAtra = new Pair(semblCA,
resultatSemblancaCicadatraAtra.getSecond());
    }

    //Cicada orni / Cicadetta cerdaniensis
    if (semblCO > 0.25 && semblCC > 0.25) {
        double semblOriginalCO = semblCO;
        double semblOriginalCC = semblCC;

        if (duradaMitjanaCantCurt > 0) {
            double coefDuradaMitjanaCant = ((50 - duradaMitjanaCantCurt) / 50) * 0.5 *
nivellConfiancaDuradaCantsCurts;
            if (coefDuradaMitjanaCant > 0) {
                semblCO /= 1 + coefDuradaMitjanaCant;
                semblCC *= 1 + coefDuradaMitjanaCant;
            }
            else {
                coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 150) / 150) * 0.8 *
nivellConfiancaDuradaCantsCurts;
                if (coefDuradaMitjanaCant > 0) {
                    semblCO *= 1 + coefDuradaMitjanaCant;
                    semblCC /= 1 + coefDuradaMitjanaCant;
                }
            }
        }
        if (duradaMitjanaSilenci > 0) {
            double coefDuradaMitjanaSilenci = Math.max(0, ((300 -
duradaMitjanaSilenci) / 300) * 0.9) * nivellConfiancaDuradaSilencis;
            if (coefDuradaMitjanaSilenci > 0) {
                semblCO *= 1 + coefDuradaMitjanaSilenci;
                semblCC /= 1 + coefDuradaMitjanaSilenci;
            }
            else {
                coefDuradaMitjanaSilenci = Math.max(0, ((duradaMitjanaSilenci - 300) /
300) * 0.6) * nivellConfiancaDuradaSilencis;
                if (coefDuradaMitjanaSilenci > 0) {
                    semblCO /= 1 + coefDuradaMitjanaSilenci;
                    semblCC *= 1 + coefDuradaMitjanaSilenci;
                }
            }
        }
        double coefProporcioSilenci = (proporcioSilenci - 0.7) * 1.2;
        if (coefProporcioSilenci > 0) {
            semblCO /= 1 + coefProporcioSilenci;
            semblCC *= 1 + coefProporcioSilenci;
        }
        else {
            semblCO *= 1 - coefProporcioSilenci;
            semblCC /= 1 - coefProporcioSilenci;
        }

        if (semblCO > 1) {
            double varAux = semblCO;
            semblCO /= varAux;
            semblCC /= varAux;
        }
        if (semblCC > 1) {
            double varAux = semblCC;
            semblCO /= varAux;
            semblCC /= varAux;
        }

        semblCO = Math.max(0, Math.min(1, semblCO));
        semblCC = Math.max(0, Math.min(1, semblCC));

        if (semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblHV > 0.25 ||
semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblCO > semblOriginalCO) {
                semblCC *= semblOriginalCO;
                semblCC /= semblCO;
                semblCO = semblOriginalCO;
            }
            else if (semblCC > semblOriginalCC) {
```

```
            semblCO *= semblOriginalCC;
            semblCO /= semblCC;
            semblCC = semblOriginalCC;
          }
        }

        resultatSemblancaCicadaOrni = new Pair(semblCO, resultatSemblancaCicadaOrni.getSecond());
        resultatSemblancaCicadettaCerdaniensis = new Pair(semblCC, resultatSemblancaCicadettaCerdaniensis.getSecond());
      }

      //Cicada orni / Hilaphura varipes
      if (semblCO > 0.25 && semblHV > 0.25) {
        double semblOriginalCO = semblCO;
        double semblOriginalHV = semblHV;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 225) / 225) * 0.6 * nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCant > 0) {
            semblCO /= 1 + coefDuradaMitjanaCant;
            semblHV *= 1 + coefDuradaMitjanaCant;
          }
          else {
            semblCO *= 1 - coefDuradaMitjanaCant;
            semblHV /= 1 - coefDuradaMitjanaCant;
          }

        }
        if (duradaMitjanaSilenci > 0) {
          double coefDuradaMitjanaSilenci = Math.max(0, ((duradaMitjanaSilenci - 170) / 170) * 0.1) * nivellConfiancaDuradaSilencis;
          if (coefDuradaMitjanaSilenci > 0) {
            semblCO /= 1 + coefDuradaMitjanaSilenci;
            semblHV *= 1 + coefDuradaMitjanaSilenci;
          }
        }
        if (frequenciaMaximaMostreig >= 11000) {
          double coefFreq = ((frequenciaMitjana + incrementFreqCO - 6625) / 6625) * 0.3;
          if (coefFreq > 0) {
            semblCO /= 1 + coefFreq;
            semblHV *= 1 + coefFreq;
          }
          else {
            semblCO *= 1 - coefFreq;
            semblHV /= 1 - coefFreq;
          }
          coefFreq = ((frequenciaMitjana + incrementFreqHV - 6625) / 6625) * 0.3;
          if (coefFreq > 0) {
            semblCO /= 1 + coefFreq;
            semblHV *= 1 + coefFreq;
          }
          else {
            semblCO *= 1 - coefFreq;
            semblHV /= 1 - coefFreq;
          }
        }

        if (semblCO > 1) {
          double varAux = semblCO;
          semblCO /= varAux;
          semblHV /= varAux;
        }
        if (semblHV > 1) {
          double varAux = semblHV;
          semblCO /= varAux;
          semblHV /= varAux;
        }

        semblCO = Math.max(0, Math.min(1, semblCO));
        semblHV = Math.max(0, Math.min(1, semblHV));

        if (semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCO > semblOriginalCO) {
            semblHV *= semblOriginalCO;
            semblHV /= semblCO;
            semblCO = semblOriginalCO;
          }
          else if (semblHV > semblOriginalHV) {
            semblCO *= semblOriginalHV;
            semblCO /= semblHV;
            semblHV = semblOriginalHV;
          }
        }

        resultatSemblancaCicadaOrni = new Pair(semblCO, resultatSemblancaCicadaOrni.getSecond());
        resultatSemblancaHilaphuraVaripes = new Pair(semblHV, resultatSemblancaHilaphuraVaripes.getSecond());
      }

      //Cicada orni / Lyristes plebejus
      if (semblCO > 0.25 && semblLP > 0.25) {
        double semblOriginalCO = semblCO;
        double semblOriginalLP = semblLP;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCantCurt = ((duradaMitjanaCantCurt - 75) / 75) * 1.25 * nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCantCurt > 0) {
            semblCO *= 1 + coefDuradaMitjanaCantCurt;
            semblLP /= 1 + coefDuradaMitjanaCantCurt;
          }
          else {
            semblCO /= 1 - coefDuradaMitjanaCantCurt;
            semblLP *= 1 - coefDuradaMitjanaCantCurt;
          }
        }
        double coefNivConfCantsCurts = ((1 - nivellConfiancaSonCantsCurts) / 1) * 0.75;
        semblCO /= 1 + coefNivConfCantsCurts;
        semblLP *= 1 + coefNivConfCantsCurts;

        if (duradaMitjanaSilenci > 0) {
          double coefDuradaMitjanaSilenci = ((duradaMitjanaSilenci - 65) / 65) * 1.25 * nivellConfiancaDuradaSilencis;
          if (coefDuradaMitjanaSilenci > 0) {
            semblCO *= 1 + coefDuradaMitjanaSilenci;
            semblLP /= 1 + coefDuradaMitjanaSilenci;
          }
          else {
            semblCO /= 1 - coefDuradaMitjanaSilenci;
            semblLP *= 1 - coefDuradaMitjanaSilenci;
          }
        }
        if (frequenciaMaximaMostreig >= 7000) {
          double coefFreq = ((frequenciaMitjana + incrementFreqCO - 6000) / 6000) * 0.3;
          if (coefFreq > 0) {
            semblCO /= 1 + coefFreq;
            semblLP *= 1 + coefFreq;
          }
          else {
            semblCO *= 1 - coefFreq;
            semblLP /= 1 - coefFreq;
          }
          coefFreq = ((frequenciaMitjana + incrementFreqLP - 6000) / 6000) * 0.3;
          if (coefFreq > 0) {
            semblCO /= 1 + coefFreq;
            semblLP *= 1 + coefFreq;
          }
          else {
            semblCO *= 1 - coefFreq;
            semblLP /= 1 - coefFreq;
          }
        }
        double coefPropSil = ((proporcioSilenci - 0.45) / 0.45) * 0.4;
        if (coefPropSil > 0) {
          semblCO *= 1 + coefPropSil;
          semblLP /= 1 + coefPropSil;
        }
        else {
          semblCO /= 1 - coefPropSil;
          semblLP *= 1 - coefPropSil;
        }

        double coefCantsLP = nivellConfiancaCantsLP * 0.25;
        semblCO /= 1 + coefCantsLP;
        semblLP *= 1 + coefCantsLP;

        if (semblCO > 1) {
          double varAux = semblCO;
          semblCO /= varAux;
          semblLP /= varAux;
        }
        if (semblLP > 1) {
          double varAux = semblLP;
          semblCO /= varAux;
          semblLP /= varAux;
        }

        semblCO = Math.max(0, Math.min(1, semblCO));
        semblLP = Math.max(0, Math.min(1, semblLP));

        if (semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCO > semblOriginalCO) {
            semblLP *= semblOriginalCO;
```

```
            semblLP /= semblCO;
            semblCO = semblOriginalCO;
          }
          else if (semblLP > semblOriginalLP) {
            semblCO *= semblOriginalLP;
            semblCO /= semblLP;
            semblLP = semblOriginalLP;
          }
        }

        resultatSemblancaCicadaOrni = new Pair(semblCO,
resultatSemblancaCicadaOrni.getSecond());
        resultatSemblancaLyristesPlebejus = new Pair(semblLP,
resultatSemblancaLyristesPlebejus.getSecond());
      }

      //Cicada orni / Tettigettalna argentata
      if (semblCO > 0.25 && semblTA > 0.25) {
        double semblOriginalCO = semblCO;
        double semblOriginalTA = semblTA;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 50) / 50) * 1.25
* nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCant > 0) {
            semblCO *= 1 + coefDuradaMitjanaCant;
            semblTA /= 1 + coefDuradaMitjanaCant;
          }
          else {
            semblCO /= 1 - coefDuradaMitjanaCant;
            semblTA *= 1 - coefDuradaMitjanaCant;
          }
        }

        double coefProporcioSilenci = (proporcioSilenci - 0.7) * 0.85;
        if (coefProporcioSilenci > 0) {
          semblCO /= 1 + coefProporcioSilenci;
          semblTA *= 1 + coefProporcioSilenci;
        }
        else {
          semblCO *= 1 - coefProporcioSilenci;
          semblTA /= 1 - coefProporcioSilenci;
        }

        if (semblCO > 1) {
          double varAux = semblCO;
          semblCO /= varAux;
          semblTA /= varAux;
        }
        if (semblTA > 1) {
          double varAux = semblTA;
          semblCO /= varAux;
          semblTA /= varAux;
        }

        semblCO = Math.max(0, Math.min(1, semblCO));
        semblTA = Math.max(0, Math.min(1, semblTA));

        if (semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCO > semblOriginalCO) {
            semblTA *= semblOriginalCO;
            semblTA /= semblCO;
            semblCO = semblOriginalCO;
          }
          else if (semblTA > semblOriginalTA) {
            semblCO *= semblOriginalTA;
            semblCO /= semblTA;
            semblTA = semblOriginalTA;
          }
        }

        resultatSemblancaCicadaOrni = new Pair(semblCO,
resultatSemblancaCicadaOrni.getSecond());
        resultatSemblancaTettigettalnaArgentata = new Pair(semblTA,
resultatSemblancaTettigettalnaArgentata.getSecond());
      }

      //Cicada orni / Tettigettula pygmea
      if (semblCO > 0.25 && semblTP > 0.25) {
        double semblOriginalCO = semblCO;
        double semblOriginalTP = semblTP;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCant = ((150 - duradaMitjanaCantCurt) / 150) * 1.7
* nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCant > 0) {
            semblCO *= 1 + coefDuradaMitjanaCant;
            semblTP /= 1 + coefDuradaMitjanaCant;
          }
          else {
            coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 250) / 250) * 0.3 *
nivellConfiancaDuradaCantsCurts;
            if (coefDuradaMitjanaCant > 0) {
              semblCO /= 1 + coefDuradaMitjanaCant;
              semblTP *= 1 + coefDuradaMitjanaCant;
            }
          }
        }

        if (semblCO > 1) {
          double varAux = semblCO;
          semblCO /= varAux;
          semblTP /= varAux;
        }
        if (semblTP > 1) {
          double varAux = semblTP;
          semblCO /= varAux;
          semblTP /= varAux;
        }

        semblCO = Math.max(0, Math.min(1, semblCO));
        semblTP = Math.max(0, Math.min(1, semblTP));

        if (semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCO > semblOriginalCO) {
            semblTP *= semblOriginalCO;
            semblTP /= semblCO;
            semblCO = semblOriginalCO;
          }
          else if (semblTP > semblOriginalTP) {
            semblCO *= semblOriginalTP;
            semblCO /= semblTP;
            semblTP = semblOriginalTP;
          }
        }

        resultatSemblancaCicadaOrni = new Pair(semblCO,
resultatSemblancaCicadaOrni.getSecond());
        resultatSemblancaTettigettulaPygmea = new Pair(semblTP,
resultatSemblancaTettigettulaPygmea.getSecond());
      }

      //Cicada orni / Tibicina haematodes
      if (semblCO > 0.25 && semblTH > 0.25) {
        double semblOriginalCO = semblCO;
        double semblOriginalTH = semblTH;


        double coefNivConfNGP = ((0.75 -
nivellConfiancaNombreGrupsDePolsosPerSegon)) * 0.8;
        if (coefNivConfNGP > 0) {
          semblCO *= 1 + coefNivConfNGP;
          semblTH /= 1 + coefNivConfNGP;
        }
        else {
          semblCO /= 1 - coefNivConfNGP;
          semblTH *= 1 - coefNivConfNGP;
        }

        if (semblCO > 1) {
          double varAux = semblCO;
          semblCO /= varAux;
          semblTH /= varAux;
        }
        if (semblTH > 1) {
          double varAux = semblTH;
          semblCO /= varAux;
          semblTH /= varAux;
        }

        semblCO = Math.max(0, Math.min(1, semblCO));
        semblTH = Math.max(0, Math.min(1, semblTH));

        if (semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC >
0.25 || semblTG > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCO > semblOriginalCO) {
            semblTH *= semblOriginalCO;
            semblTH /= semblCO;
            semblCO = semblOriginalCO;
          }
          else if (semblTH > semblOriginalTH) {
            semblCO *= semblOriginalTH;
            semblCO /= semblTH;
            semblTH = semblOriginalTH;
          }
        }
```

```
        resultatSemblancaCicadaOrni = new Pair(semblCO, resultatSemblancaCicadaOrni.getSecond());
        resultatSemblancaTibicinaHaematodes = new Pair(semblTH, resultatSemblancaTibicinaHaematodes.getSecond());
    }

    //Cicada barbara lusitanica / Cicadetta cerdaniensis
    if (semblCB > 0.25 && semblCC > 0.25) {
        double semblOriginalCB = semblCB;
        double semblOriginalCC = semblCC;


        double coefBatecs = Math.max(0, nBatecsAlesOCantsIncorrectes * 0.1);
        if (coefBatecs > 0) {
            semblCB *= 1 + coefBatecs;
            semblCC /= 1 + coefBatecs;
        }

        double coefNivConfDurCantCurt = Math.max(0, (0.5 - nivellConfiancaDuradaCantsCurts) * 0.75);
        if (coefNivConfDurCantCurt > 0) {
            semblCB *= 1 + coefNivConfDurCantCurt;
            semblCC /= 1 + coefNivConfDurCantCurt;
        }

        double coefNivConfEsCantLlargIrreg = (nivellConfiancaEsCantLlargIrregular - 0.4) * 0.5;
        if (coefNivConfEsCantLlargIrreg > 0) {
            semblCB *= 1 + coefNivConfEsCantLlargIrreg;
            semblCC /= 1 + coefNivConfEsCantLlargIrreg;
        }

        if (semblCB > 1) {
            double varAux = semblCB;
            semblCB /= varAux;
            semblCC /= varAux;
        }
        if (semblCC > 1) {
            double varAux = semblCC;
            semblCB /= varAux;
            semblCC /= varAux;
        }

        semblCB = Math.max(0, Math.min(1, semblCB));
        semblCC = Math.max(0, Math.min(1, semblCC));

        if (semblCO > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblCB > semblOriginalCB) {
                semblCC *= semblOriginalCB;
                semblCC /= semblCB;
                semblCB = semblOriginalCB;
            }
            else if (semblCC > semblOriginalCC) {
                semblCB *= semblOriginalCC;
                semblCB /= semblCC;
                semblCC = semblOriginalCC;
            }
        }

        resultatSemblancaCicadaBarbaraLusitanica = new Pair(semblCB, resultatSemblancaCicadaBarbaraLusitanica.getSecond());
        resultatSemblancaCicadettaCerdaniensis = new Pair(semblCC, resultatSemblancaCicadettaCerdaniensis.getSecond());
    }

    //Cicada barbara lusitanica / Lyristes plebejus
    if (semblCB > 0.25 && semblLP > 0.25) {
        double semblOriginalCB = semblCB;
        double semblOriginalLP = semblLP;

        if (frequenciaMaximaMostreig >= 9000) {
            double coefFreq = ((frequenciaMitjana + incrementFreqCB - 7000) / 7000) * 0.15;
            if (coefFreq > 0) {
                semblCB /= 1 + coefFreq;
                semblLP *= 1 + coefFreq;
            }
            coefFreq = ((frequenciaMitjana + incrementFreqLP - 7000) / 7000) * 0.15;
            if (coefFreq > 0) {
                semblCB /= 1 + coefFreq;
                semblLP *= 1 + coefFreq;
            }
        }

        if (nivellConfiancaSonCantsLlargs < 0.25) {
            double coefCantsLlargs = (0.25 - nivellConfiancaSonCantsLlargs) * 0.15;
            semblCB /= 1 + coefCantsLlargs;
            semblLP *= 1 + coefCantsLlargs;
        }
        else if (nivellConfiancaSonCantsLlargs > 0.75) {
            double coefCantsLlargs = (nivellConfiancaSonCantsLlargs - 0.75) * 0.1;
            semblCB *= 1 + coefCantsLlargs;
            semblLP /= 1 + coefCantsLlargs;
        }

        double coefCantsLP = nivellConfiancaCantsLP * 7.5;
        semblCB /= 1 + coefCantsLP;
        semblLP *= 1 + coefCantsLP;

        if (semblCB > 1) {
            double varAux = semblCB;
            semblCB /= varAux;
            semblLP /= varAux;
        }
        if (semblLP > 1) {
            double varAux = semblLP;
            semblCB /= varAux;
            semblLP /= varAux;
        }


        semblCB = Math.max(0, Math.min(1, semblCB));
        semblLP = Math.max(0, Math.min(1, semblLP));

        if (semblCO > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblCB > semblOriginalCB) {
                semblLP *= semblOriginalCB;
                semblLP /= semblCB;
                semblCB = semblOriginalCB;
            }
            else if (semblLP > semblOriginalLP) {
                semblCB *= semblOriginalLP;
                semblCB /= semblLP;
                semblLP = semblOriginalLP;
            }
        }

        resultatSemblancaCicadaBarbaraLusitanica = new Pair(semblCB, resultatSemblancaCicadaBarbaraLusitanica.getSecond());
        resultatSemblancaLyristesPlebejus = new Pair(semblLP, resultatSemblancaLyristesPlebejus.getSecond());
    }

    //Cicada barbara lusitanica / Tettigettula pygmea
    if (semblCB > 0.25 && semblTP > 0.25) {
        double semblOriginalCB = semblCB;
        double semblOriginalTP = semblTP;

        if (nivellConfiancaSonCantsLlargs < 0.25) {
            double coefCantsLlargs = (0.25 - nivellConfiancaSonCantsLlargs) * 0.15;
            semblCB /= 1 + coefCantsLlargs;
            semblTP *= 1 + coefCantsLlargs;
        }
        else if (nivellConfiancaSonCantsLlargs > 0.75) {
            double coefCantsLlargs = (nivellConfiancaSonCantsLlargs - 0.75) * 0.15;
            semblCB *= 1 + coefCantsLlargs;
            semblTP /= 1 + coefCantsLlargs;
        }

        double coefNivConfDurCantCurt = Math.max(0, (0.6 - nivellConfiancaDuradaCantsCurts) * 0.85);
        if (coefNivConfDurCantCurt > 0) {
            semblCB *= 1 + coefNivConfDurCantCurt;
            semblTP /= 1 + coefNivConfDurCantCurt;
        }

        double coefNivConfEsCantLlargIrreg = (nivellConfiancaEsCantLlargIrregular - 0.4) * 0.9;
        if (coefNivConfEsCantLlargIrreg > 0) {
            semblCB *= 1 + coefNivConfEsCantLlargIrreg;
            semblTP /= 1 + coefNivConfEsCantLlargIrreg;
        }

        if (semblCB > 1) {
            double varAux = semblCB;
            semblCB /= varAux;
            semblTP /= varAux;
        }
        if (semblTP > 1) {
            double varAux = semblTP;
            semblCB /= varAux;
            semblTP /= varAux;
        }

        semblCB = Math.max(0, Math.min(1, semblCB));
        semblTP = Math.max(0, Math.min(1, semblTP));
```

```
        if (semblCO > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCB > semblOriginalCB) {
            semblTP *= semblOriginalCB;
            semblTP /= semblCB;
            semblCB = semblOriginalCB;
          }
          else if (semblTP > semblOriginalTP) {
            semblCB *= semblOriginalTP;
            semblCB /= semblTP;
            semblTP = semblOriginalTP;
          }
        }

        resultatSemblancaCicadaBarbaraLusitanica = new Pair(semblCB,
resultatSemblancaCicadaBarbaraLusitanica.getSecond());
        resultatSemblancaTettigettulaPygmea = new Pair(semblTP,
resultatSemblancaTettigettulaPygmea.getSecond());
      }

      //Cicada barbara lusitanica / Tibicina garricola
      if (semblCB > 0.25 && semblTG > 0.25) {
        double semblOriginalCB = semblCB;
        double semblOriginalTG = semblTG;


        double coefNivConfNGP = ((0.7 -
nivellConfiancaNombreGrupsDePolsosPerSegon)) * 0.8;
        if (coefNivConfNGP > 0) {
          semblCB *= 1 + coefNivConfNGP;
          semblTG /= 1 + coefNivConfNGP;
        }
        else {
          semblCB /= 1 - coefNivConfNGP;
          semblTG *= 1 - coefNivConfNGP;
        }

        double coefNivConfEsCantLlargIrreg = (nivellConfiancaEsCantLlargIrregular -
0.4) * 0.5;
        if (coefNivConfEsCantLlargIrreg > 0) {
          semblCB *= 1 + coefNivConfEsCantLlargIrreg;
          semblTG /= 1 + coefNivConfEsCantLlargIrreg;
        }

        if (semblCB > 1) {
          double varAux = semblCB;
          semblCB /= varAux;
          semblTG /= varAux;
        }
        if (semblTG > 1) {
          double varAux = semblTG;
          semblCB /= varAux;
          semblTG /= varAux;
        }

        semblCB = Math.max(0, Math.min(1, semblCB));
        semblTG = Math.max(0, Math.min(1, semblTG));

        if (semblCO > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCB > semblOriginalCB) {
            semblTG *= semblOriginalCB;
            semblTG /= semblCB;
            semblCB = semblOriginalCB;
          }
          else if (semblTG > semblOriginalTG) {
            semblCB *= semblOriginalTG;
            semblCB /= semblTG;
            semblTG = semblOriginalTG;
          }
        }

        resultatSemblancaCicadaBarbaraLusitanica = new Pair(semblCB,
resultatSemblancaCicadaBarbaraLusitanica.getSecond());
        resultatSemblancaTibicinaGarricola = new Pair(semblTG,
resultatSemblancaTibicinaGarricola.getSecond());
      }

      //Cicada barbara lusitanica / Tibicina haematodes
      if (semblCB > 0.25 && semblTH > 0.25) {
        double semblOriginalCB = semblCB;
        double semblOriginalTH = semblTH;


        double coefNivConfNGP = ((0.7 -
nivellConfiancaNombreGrupsDePolsosPerSegon)) * 0.8;
        if (coefNivConfNGP > 0) {
          semblCB *= 1 + coefNivConfNGP;
          semblTH /= 1 + coefNivConfNGP;
        }
        else {
          semblCB /= 1 - coefNivConfNGP;
          semblTH *= 1 - coefNivConfNGP;
        }

        double coefNivConfEsCantLlargIrreg = (nivellConfiancaEsCantLlargIrregular -
0.4) * 0.5;
        if (coefNivConfEsCantLlargIrreg > 0) {
          semblCB *= 1 + coefNivConfEsCantLlargIrreg;
          semblLP /= 1 + coefNivConfEsCantLlargIrreg;
        }

        if (semblCB > 1) {
          double varAux = semblCB;
          semblCB /= varAux;
          semblTH /= varAux;
        }
        if (semblTH > 1) {
          double varAux = semblTH;
          semblCB /= varAux;
          semblTH /= varAux;
        }

        semblCB = Math.max(0, Math.min(1, semblCB));
        semblTH = Math.max(0, Math.min(1, semblTH));

        if (semblCO > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC >
0.25 || semblTG > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCB > semblOriginalCB) {
            semblTH *= semblOriginalCB;
            semblTH /= semblCB;
            semblCB = semblOriginalCB;
          }
          else if (semblTH > semblOriginalTH) {
            semblCB *= semblOriginalTH;
            semblCB /= semblTH;
            semblTH = semblOriginalTH;
          }
        }

        resultatSemblancaCicadaBarbaraLusitanica = new Pair(semblCB,
resultatSemblancaCicadaBarbaraLusitanica.getSecond());
        resultatSemblancaTibicinaHaematodes = new Pair(semblTH,
resultatSemblancaTibicinaHaematodes.getSecond());
      }

      //Cicadatra atra / Cicadetta cerdaniensis
      if (semblCA > 0.25 && semblCC > 0.25) {
        double semblOriginalCA = semblCA;
        double semblOriginalCC = semblCC;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 140) / 140) * 0.8
* nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCant > 0) {
            semblCA /= 1 + coefDuradaMitjanaCant;
            semblCC *= 1 + coefDuradaMitjanaCant;
          }
          else {
            semblCA *= 1 - coefDuradaMitjanaCant;
            semblCC /= 1 - coefDuradaMitjanaCant;
          }
        }
        if (duradaMitjanaSilenci > 0) {
          double coefDuradaMitjanaSilenci = ((170 - duradaMitjanaSilenci) / 170) * 0.9
* nivellConfiancaDuradaSilencis;
          if (coefDuradaMitjanaSilenci > 0) {
            semblCA *= 1 + coefDuradaMitjanaSilenci;
            semblCC /= 1 + coefDuradaMitjanaSilenci;
          }
          else {
            semblCA /= 1 - coefDuradaMitjanaSilenci;
            semblCC *= 1 - coefDuradaMitjanaSilenci;
          }
        }

        if (frequenciaMaximaMostreig >= 11000) {
          double coefFreq =((frequenciaMitjana + incrementFreqCA - 13000) / 13000) *
0.6;
          if (coefFreq > 0) {
            semblCA /= 1 + coefFreq;
            semblCC *= 1 + coefFreq;
          }
          else {
            semblCA *= 1 - coefFreq;
            semblCC /= 1 - coefFreq;
          }
          coefFreq =((frequenciaMitjana + incrementFreqCC - 13000) / 13000) * 0.6;
          if (coefFreq > 0) {
```

```
            semblCA /= 1 + coefFreq;
            semblCC *= 1 + coefFreq;
          }
          else {
            semblCA *= 1 - coefFreq;
            semblCC /= 1 - coefFreq;
          }
        }

        if (semblCA > 1) {
          double varAux = semblCA;
          semblCA /= varAux;
          semblCC /= varAux;
        }
        if (semblCC > 1) {
          double varAux = semblCC;
          semblCA /= varAux;
          semblCC /= varAux;
        }

        semblCA = Math.max(0, Math.min(1, semblCA));
        semblCC = Math.max(0, Math.min(1, semblCC));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCBr > 0.25 || semblHV > 0.25 ||
semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCA > semblOriginalCA) {
            semblCC *= semblOriginalCA;
            semblCC /= semblCA;
            semblCA = semblOriginalCA;
          }
          else if (semblCC > semblOriginalCC) {
            semblCA *= semblOriginalCC;
            semblCA /= semblCC;
            semblCC = semblOriginalCC;
          }
        }

        resultatSemblancaCicadatraAtra = new Pair(semblCA,
resultatSemblancaCicadatraAtra.getSecond());
        resultatSemblancaCicadettaCerdaniensis = new Pair(semblCC,
resultatSemblancaCicadettaCerdaniensis.getSecond());
      }

      //Cicadatra atra / Lyristes plebejus
      if (semblCA > 0.25 && semblLP > 0.25) {
        double semblOriginalCA = semblCA;
        double semblOriginalLP = semblLP;

        if (duradaMitjanaCantCurt > 0) {
          if (duradaMitjanaCantCurt < 100 && nivellConfiancaDuradaCantsCurts > 0.5)
semblCA *= 0;
          else {
            double coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 100) / 100) *
0.75 * nivellConfiancaDuradaCantsCurts;
            if (coefDuradaMitjanaCant > 0) {
              semblCA *= 1 + coefDuradaMitjanaCant;
              semblLP /= 1 + coefDuradaMitjanaCant;
            }
            else {
              semblCA /= 1 - coefDuradaMitjanaCant;
              semblLP *= 1 - coefDuradaMitjanaCant;
            }
          }
        }

        if (duradaMitjanaSilenci > 0) {
          double coefDuradaMitjanaSilenci = ((duradaMitjanaSilenci - 80) / 80) * 0.7 *
nivellConfiancaDuradaSilencis;
          if (coefDuradaMitjanaSilenci > 0) {
            semblCA *= 1 + coefDuradaMitjanaSilenci;
            semblLP /= 1 + coefDuradaMitjanaSilenci;
          }
          else {
            semblCA /= 1 - coefDuradaMitjanaSilenci;
            semblLP *= 1 - coefDuradaMitjanaSilenci;
          }
        }

        if (frequenciaMaximaMostreig >= 11000) {
          double coefFreq =((frequenciaMitjana + incrementFreqCA - 9750) / 9750) *
0.9;
          if (coefFreq > 0) {
            semblCA *= 1 + coefFreq;
            semblLP /= 1 + coefFreq;
          }
          else {
            semblCA /= 1 - coefFreq;
            semblLP *= 1 - coefFreq;
          }
          coefFreq =((frequenciaMitjana + incrementFreqLP - 9750) / 9750) * 0.9;
          if (coefFreq > 0) {
            semblCA *= 1 + coefFreq;
            semblLP /= 1 + coefFreq;
          }
          else {
            semblCA /= 1 - coefFreq;
            semblLP *= 1 - coefFreq;
          }
        }

        double coefCantsLP = nivellConfiancaCantsLP * 5.5;
        semblCA /= 1 + coefCantsLP;
        semblLP *= 1 + coefCantsLP;

        if (semblCA > 1) {
          double varAux = semblCA;
          semblCA /= varAux;
          semblLP /= varAux;
        }
        if (semblLP > 1) {
          double varAux = semblLP;
          semblCA /= varAux;
          semblLP /= varAux;
        }

        semblCA = Math.max(0, Math.min(1, semblCA));
        semblLP = Math.max(0, Math.min(1, semblLP));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCA > semblOriginalCA) {
            semblLP *= semblOriginalCA;
            semblLP /= semblCA;
            semblCA = semblOriginalCA;
          }
          else if (semblLP > semblOriginalLP) {
            semblCA *= semblOriginalLP;
            semblCA /= semblLP;
            semblLP = semblOriginalLP;
          }
        }

        resultatSemblancaCicadatraAtra = new Pair(semblCA,
resultatSemblancaCicadatraAtra.getSecond());
        resultatSemblancaLyristesPlebejus = new Pair(semblLP,
resultatSemblancaLyristesPlebejus.getSecond());
      }

      //Cicadatra atra / Tettigettula pygmea
      if (semblCA > 0.25 && semblTP > 0.25) {
        double semblOriginalCA = semblCA;
        double semblOriginalTP = semblTP;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCant = ((200 - duradaMitjanaCantCurt) / 200) *
0.15 * nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCant > 0) {
            semblCA /= 1 + coefDuradaMitjanaCant;
            semblTP *= 1 + coefDuradaMitjanaCant;
          }
          else {
            coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 280) / 280) * 0.9 *
nivellConfiancaDuradaCantsCurts;
            if (coefDuradaMitjanaCant > 0) {
              semblCA *= 1 + coefDuradaMitjanaCant;
              semblTP /= 1 + coefDuradaMitjanaCant;
            }
          }
        }

        if (duradaMitjanaSilenci > 0) {
          double coefDuradaMitjanaSilenci = ((100 - duradaMitjanaSilenci) / 100) * 0.2;
          if (coefDuradaMitjanaSilenci > 0) {
            semblCA /= 1 + coefDuradaMitjanaSilenci;
            semblTP *= 1 + coefDuradaMitjanaSilenci;
          }
          else {
            coefDuradaMitjanaSilenci = Math.min(((duradaMitjanaSilenci - 160) / 160)
* 1.3, 0.25);
            if (coefDuradaMitjanaSilenci > 0) {
              semblCA *= 1 + coefDuradaMitjanaSilenci;
              semblTP /= 1 + coefDuradaMitjanaSilenci;
            }
          }
        }

        if (semblCA > 1) {
          double varAux = semblCA;
          semblCA /= varAux;
          semblTP /= varAux;
```

```
        }
        if (semblTP > 1) {
          double varAux = semblTP;
          semblCA /= varAux;
          semblTP /= varAux;
        }

        semblCA = Math.max(0, Math.min(1, semblCA));
        semblTP = Math.max(0, Math.min(1, semblTP));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCA > semblOriginalCA) {
            semblTP *= semblOriginalCA;
            semblTP /= semblCA;
            semblCA = semblOriginalCA;
          }
          else if (semblTP > semblOriginalTP) {
            semblCA *= semblOriginalTP;
            semblCA /= semblTP;
            semblTP = semblOriginalTP;
          }
        }

        resultatSemblancaCicadatraAtra = new Pair(semblCA,
resultatSemblancaCicadatraAtra.getSecond());
        resultatSemblancaTettigettulaPygmea = new Pair(semblTP,
resultatSemblancaTettigettulaPygmea.getSecond());
      }

      //Cicadatra atra / Tibicina corsica fairmairei
      if (semblCA > 0.25 && semblTC > 0.25) {
        double semblOriginalCA = semblCA;
        double semblOriginalTC = semblTC;

        if (duradaMitjanaCantCurt > 0) {
          double coefMitjDurCantCurt = ((duradaMitjanaCantCurt - 200)) * 0.3 *
nivellConfiancaDuradaCantsCurts;
          if (coefMitjDurCantCurt > 0) {
            semblCA *= 1 + coefMitjDurCantCurt;
            semblTC /= 1 + coefMitjDurCantCurt;
          }
          else {
            semblCA /= 1 - coefMitjDurCantCurt;
            semblTC *= 1 - coefMitjDurCantCurt;
          }
        }

        if (duradaMitjanaSilenci > 0) {
          double coefMitjDurSil = ((duradaMitjanaSilenci - 125) / 125) * 0.2;
          if (coefMitjDurSil > 0) {
            semblCA *= 1 + coefMitjDurSil;
            semblTC /= 1 + coefMitjDurSil;
          }
          else {
            semblCA /= 1 - coefMitjDurSil;
            semblTC *= 1 - coefMitjDurSil;
          }
        }

        double coefNivConfNGP = ((nivellConfiancaNombreGrupsDePolsosPerSegon -
0.7)) * 0.6;
        if (coefNivConfNGP > 0) {
          semblCA /= 1 + coefNivConfNGP;
          semblTC *= 1 + coefNivConfNGP;
        }
        else {
          semblCA *= 1 - coefNivConfNGP;
          semblTC /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((59 - nombreGrupsDePolsosPerSegon) / 59) * 3.5 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblCA *= 1 + coefNGP;
            semblTC /= 1 + coefNGP;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 65) / 65) * 3.5 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblCA *= 1 + coefNGP;
              semblTC /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 59 &&
nombreGrupsDePolsosPerSegon <= 65) {
              semblCA /= 1.25;
              semblTC *= 1.25;
            }
          }
        }

        if (semblCA > 1) {
          double varAux = semblCA;
          semblCA /= varAux;
          semblTC /= varAux;
        }
        if (semblTC > 1) {
          double varAux = semblTC;
          semblCA /= varAux;
          semblTC /= varAux;
        }

        semblCA = Math.max(0, Math.min(1, semblCA));
        semblTC = Math.max(0, Math.min(1, semblTC));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCA > semblOriginalCA) {
            semblTC *= semblOriginalCA;
            semblTC /= semblCA;
            semblCA = semblOriginalCA;
          }
          else if (semblTC > semblOriginalTC) {
            semblCA *= semblOriginalTC;
            semblCA /= semblTC;
            semblTC = semblOriginalTC;
          }
        }

        resultatSemblancaCicadatraAtra = new Pair(semblCA,
resultatSemblancaCicadatraAtra.getSecond());
        resultatSemblancaTibicinaCorsicaFairmairei = new Pair(semblTC,
resultatSemblancaTibicinaCorsicaFairmairei.getSecond());
      }

      //Cicadatra atra / Tibicina garricola
      if (semblCA > 0.25 && semblTG > 0.25) {
        double semblOriginalCA = semblCA;
        double semblOriginalTG = semblTG;

        if (duradaMitjanaCantCurt > 0) {
          double coefMitjDurCantCurt = ((duradaMitjanaCantCurt - 200)) * 0.3 *
nivellConfiancaDuradaCantsCurts;
          if (coefMitjDurCantCurt > 0) {
            semblCA *= 1 + coefMitjDurCantCurt;
            semblTG /= 1 + coefMitjDurCantCurt;
          }
          else {
            semblCA /= 1 - coefMitjDurCantCurt;
            semblTG *= 1 - coefMitjDurCantCurt;
          }
        }

        if (duradaMitjanaSilenci > 0) {
          double coefMitjDurSil = ((duradaMitjanaSilenci - 125) / 125) * 0.2;
          if (coefMitjDurSil > 0) {
            semblCA *= 1 + coefMitjDurSil;
            semblTG /= 1 + coefMitjDurSil;
          }
          else {
            semblCA /= 1 - coefMitjDurSil;
            semblTG *= 1 - coefMitjDurSil;
          }
        }

        double coefNivConfNGP = (nivellConfiancaNombreGrupsDePolsosPerSegon -
0.7) * 0.5;
        if (coefNivConfNGP > 0) {
          semblCA /= 1 + coefNivConfNGP;
          semblTG *= 1 + coefNivConfNGP;
        }
        else {
          semblCA *= 1 - coefNivConfNGP;
          semblTG /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((66 - nombreGrupsDePolsosPerSegon) / 66) * 3.25 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblCA *= 1 + coefNGP;
            semblTG /= 1 + coefNGP;
          }
          else {
            if (nombreGrupsDePolsosPerSegon > 132 &&
nombreGrupsDePolsosPerSegon < 142) {
              semblCA *= 1.2;
              semblTG /= 1.2;
```

```
            }
            else {
              coefNGP = ((nombreGrupsDePolsosPerSegon - 71) / 71) * 3.25 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
              if (coefNGP > 0) {
                semblCA *= 1 + coefNGP;
                semblTG /= 1 + coefNGP;
              }
              else if (nombreGrupsDePolsosPerSegon >= 66 &&
nombreGrupsDePolsosPerSegon <= 71) {
                semblCA /= 1.25;
                semblTG *= 1.25;
              }
            }
          }
        }

        double coefFreq = ((frequenciaMitjana + incrementFreqCA - 9750) / 9750) *
0.65;
        if (coefFreq > 0) {
          semblCA *= 1 + coefFreq;
          semblTG /= 1 + coefFreq;
        }
        else {
          semblCA /= 1 - coefFreq;
          semblTG *= 1 - coefFreq;
        }
        coefFreq = ((frequenciaMitjana + incrementFreqTG - 9750) / 9750) * 0.65;
        if (coefFreq > 0) {
          semblCA *= 1 + coefFreq;
          semblTG /= 1 + coefFreq;
        }
        else {
          semblCA /= 1 - coefFreq;
          semblTG *= 1 - coefFreq;
        }

        if (semblCA > 1) {
          double varAux = semblCA;
          semblCA /= varAux;
          semblTG /= varAux;
        }
        if (semblTG > 1) {
          double varAux = semblTG;
          semblCA /= varAux;
          semblTG /= varAux;
        }

        semblCA = Math.max(0, Math.min(1, semblCA));
        semblTG = Math.max(0, Math.min(1, semblTG));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCA > semblOriginalCA) {
            semblTG *= semblOriginalCA;
            semblTG /= semblCA;
            semblCA = semblOriginalCA;
          }
          else if (semblTG > semblOriginalTG) {
            semblCA *= semblOriginalTG;
            semblCA /= semblTG;
            semblTG = semblOriginalTG;
          }
        }

        resultatSemblancaCicadatraAtra = new Pair(semblCA,
resultatSemblancaCicadatraAtra.getSecond());
        resultatSemblancaTibicinaGarricola = new Pair(semblTG,
resultatSemblancaTibicinaGarricola.getSecond());
      }

      //Cicadatra atra / Tibicina haematodes
      if (semblCA > 0.25 && semblTH > 0.25) {
        double semblOriginalCA = semblCA;
        double semblOriginalTH = semblTH;

        if (percentatgeCantsCurts == 100) {
          semblCA *= 1.1;
          semblTH /= 1.1;
        }

        double coefNivConfNGP = ((nivellConfiancaNombreGrupsDePolsosPerSegon -
0.75)) * 0.8;
        if (coefNivConfNGP > 0) {
          semblCA /= 1 + coefNivConfNGP;
          semblTH *= 1 + coefNivConfNGP;
        }
        else {
          semblCA *= 1 - coefNivConfNGP;
          semblTH /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((90 - nombreGrupsDePolsosPerSegon) / 90) * 3.0 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblCA *= 1 + coefNGP;
            semblTH /= 1 + coefNGP;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 105) / 105) * 3.0 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblCA *= 1 + coefNGP;
              semblTH /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 90 &&
nombreGrupsDePolsosPerSegon <= 105) {
              semblCA /= 1.25;
              semblTH *= 1.25;
            }
          }
        }

        if (semblCA > 1) {
          double varAux = semblCA;
          semblCA /= varAux;
          semblTH /= varAux;
        }
        if (semblTH > 1) {
          double varAux = semblTH;
          semblCA /= varAux;
          semblTH /= varAux;
        }

        semblCA = Math.max(0, Math.min(1, semblCA));
        semblTH = Math.max(0, Math.min(1, semblTH));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCBr > 0.25 || semblCC > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC >
0.25 || semblTG > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCA > semblOriginalCA) {
            semblTH *= semblOriginalCA;
            semblTH /= semblCA;
            semblCA = semblOriginalCA;
          }
          else if (semblTH > semblOriginalTH) {
            semblCA *= semblOriginalTH;
            semblCA /= semblTH;
            semblTH = semblOriginalTH;
          }
        }

        resultatSemblancaCicadatraAtra = new Pair(semblCA,
resultatSemblancaCicadatraAtra.getSecond());
        resultatSemblancaTibicinaHaematodes = new Pair(semblTH,
resultatSemblancaTibicinaHaematodes.getSecond());
      }

      //Cicadatra atra / Tibicina quadrisignata
      if (semblCA > 0.25 && semblTQ > 0.25) {
        double semblOriginalCA = semblCA;
        double semblOriginalTQ = semblTQ;

        semblCA /= 1 + Math.max(0, (pendentPrimeraMeitatCantsLlargs - 1.1) * 0.6);
        semblTQ *= 1 + Math.max(0, (pendentPrimeraMeitatCantsLlargs - 1.1) * 0.6);

        semblCA /= 1 + Math.max(0, (pendentSegonaMeitatCantsLlargs - 1.1) * 0.6);
        semblTQ *= 1 + Math.max(0, (pendentSegonaMeitatCantsLlargs - 1.1) * 0.6);

        if (frequenciaMaximaMostreig >= 11000) {
          double coefFreq = ((frequenciaMitjana + incrementFreqCA - 9750) / 9750) *
0.8;
          if (coefFreq > 0) {
            semblCA *= 1 + coefFreq;
            semblTQ /= 1 + coefFreq;
          }
          else {
            semblCA /= 1 - coefFreq;
            semblTQ *= 1 - coefFreq;
          }
          coefFreq = ((frequenciaMitjana + incrementFreqTQ - 9750) / 9750) * 0.8;
          if (coefFreq > 0) {
            semblCA *= 1 + coefFreq;
            semblTQ /= 1 + coefFreq;
          }
          else {
            semblCA /= 1 - coefFreq;
            semblTQ *= 1 - coefFreq;
          }
        }
```

```
        double coefNivConfNGP = ((nivellConfiancaNombreGrupsDePolsosPerSegon - 0.7)) * 0.6;
        if (coefNivConfNGP > 0) {
          semblCA /= 1 + coefNivConfNGP;
          semblTQ *= 1 + coefNivConfNGP;
        }
        else {
          semblCA *= 1 - coefNivConfNGP;
          semblTQ /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((73 - nombreGrupsDePolsosPerSegon) / 73) * 3.0 * nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblCA *= 1 + coefNGP;
            semblTQ /= 1 + coefNGP;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 80) / 80) * 3.0 * nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblCA *= 1 + coefNGP;
              semblTQ /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 73 && nombreGrupsDePolsosPerSegon <= 80) {
              semblCA /= 1.25;
              semblTQ *= 1.25;
            }
          }
        }

        semblCA /= 1 + Math.max(0, (proporcioSilenci - 0.3) * 0.3);
        semblTQ *= 1 + Math.max(0, (proporcioSilenci - 0.3) * 0.3);

        if (semblCA > 1) {
          double varAux = semblCA;
          semblCA /= varAux;
          semblTQ /= varAux;
        }
        if (semblTQ > 1) {
          double varAux = semblTQ;
          semblCA /= varAux;
          semblTQ /= varAux;
        }

        semblCA = Math.max(0, Math.min(1, semblCA));
        semblTQ = Math.max(0, Math.min(1, semblTQ));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTT > 0.25) {
          if (semblCA > semblOriginalCA) {
            semblTQ *= semblOriginalCA;
            semblTQ /= semblCA;
            semblCA = semblOriginalCA;
          }
          else if (semblTQ > semblOriginalTQ) {
            semblCA *= semblOriginalTQ;
            semblCA /= semblTQ;
            semblTQ = semblOriginalTQ;
          }
        }

        resultatSemblancaCicadatraAtra = new Pair(semblCA, resultatSemblancaCicadatraAtra.getSecond());
        resultatSemblancaTibicinaQuadrisignata = new Pair(semblTQ, resultatSemblancaTibicinaQuadrisignata.getSecond());
      }

      //Cicadetta brevipennis / Tettigettula pygmea
      if (semblCBr > 0.25 && semblTP > 0.25) {
        double semblOriginalCBr = semblCBr;
        double semblOriginalTP = semblTP;

        if (duradaMitjanaCantLlargAmbCantCurtPosterior > 0) {
          double coefDuradaMitjanaCantLlargAmbCantCurtPost = ((duradaMitjanaCantLlargAmbCantCurtPosterior - 650) / 650) * 25;
          if (coefDuradaMitjanaCantLlargAmbCantCurtPost > 0) {
            semblCBr *= 1 + coefDuradaMitjanaCantLlargAmbCantCurtPost;
            semblTP /= 1 + coefDuradaMitjanaCantLlargAmbCantCurtPost;
          }
          else {
            semblCBr /= 1 - coefDuradaMitjanaCantLlargAmbCantCurtPost;
            semblTP *= 1 - coefDuradaMitjanaCantLlargAmbCantCurtPost;
          }
        }

        if (semblCBr > 1) {
          double varAux = semblCBr;
          semblCBr /= varAux;
          semblTP /= varAux;
        }
        if (semblTP > 1) {
          double varAux = semblTP;
          semblCBr /= varAux;
          semblTP /= varAux;
        }

        semblCBr = Math.max(0, Math.min(1, semblCBr));
        semblTP = Math.max(0, Math.min(1, semblTP));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCBr > semblOriginalCBr) {
            semblTP *= semblOriginalCBr;
            semblTP /= semblCBr;
            semblCBr = semblOriginalCBr;
          }
          else if (semblTP > semblOriginalTP) {
            semblCBr *= semblOriginalTP;
            semblCBr /= semblTP;
            semblTP = semblOriginalTP;
          }
        }

        resultatSemblancaCicadettaBrevipennis = new Pair(semblCBr, resultatSemblancaCicadettaBrevipennis.getSecond());
        resultatSemblancaTettigettulaPygmea = new Pair(semblTP, resultatSemblancaTettigettulaPygmea.getSecond());
      }

      //Cicadetta cerdaniensis / Tettigettalna argentata
      if (semblCC > 0.25 && semblTA > 0.25) {
        double semblOriginalCC = semblCC;
        double semblOriginalTA = semblTA;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCant = ((40 - duradaMitjanaCantCurt) / 40) * 2.25 * nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCant > 0) {
            semblCC /= 1 + coefDuradaMitjanaCant;
            semblTA *= 1 + coefDuradaMitjanaCant;
          }
          else {
            coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 50) / 50) * 0.85 * nivellConfiancaDuradaCantsCurts;
            if (coefDuradaMitjanaCant > 0) {
              semblCC *= 1 + coefDuradaMitjanaCant;
              semblTA /= 1 + coefDuradaMitjanaCant;
            }
          }
        }

        if (duradaMitjanaSilenci > 0) {
          double coefDuradaMitjanaSilenci = ((duradaMitjanaSilenci - 140) / 140) * 0.9;
          if (coefDuradaMitjanaSilenci > 0) {
            semblCC *= 1 + coefDuradaMitjanaSilenci;
            semblTA /= 1 + coefDuradaMitjanaSilenci;
          }
          else {
            semblCC /= 1 - coefDuradaMitjanaSilenci;
            semblTA *= 1 - coefDuradaMitjanaSilenci;
          }
        }
        if (frequenciaMaximaMostreig >= 11000) {
          double coefMitjFreq = ((frequenciaMitjana + incrementFreqCC - 13250) / 13250) * 0.25;
          if (coefMitjFreq > 0) {
            semblCC *= 1 + coefMitjFreq;
            semblTA /= 1 + coefMitjFreq;
          }
          else {
            semblCC /= 1 - coefMitjFreq;
            semblTA *= 1 - coefMitjFreq;
          }
          coefMitjFreq = ((frequenciaMitjana + incrementFreqTA - 13250) / 13250) * 0.25;
          if (coefMitjFreq > 0) {
            semblCC *= 1 + coefMitjFreq;
            semblTA /= 1 + coefMitjFreq;
          }
          else {
            semblCC /= 1 - coefMitjFreq;
            semblTA *= 1 - coefMitjFreq;
          }
        }

        if (proporcioSilenci > 0) {
```

```
            double coefPropSil = (proporcioSilenci - 0.9) * 2.5;
            if (coefPropSil > 0) {
               semblCC *= 1 + coefPropSil;
               semblTA /= 1 + coefPropSil;
            }
            else {
               coefPropSil = (0.6 - proporcioSilenci) * 2.5;
               if (coefPropSil > 0) {
                  semblCC /= 1 + coefPropSil;
                  semblTA *= 1 + coefPropSil;
               }
            }
         }

         if (semblCC > 1) {
            double varAux = semblCC;
            semblCC /= varAux;
            semblTA /= varAux;
         }
         if (semblTA > 1) {
            double varAux = semblTA;
            semblCC /= varAux;
            semblTA /= varAux;
         }

         semblCC = Math.max(0, Math.min(1, semblCC));
         semblTA = Math.max(0, Math.min(1, semblTA));

         if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblCC > semblOriginalCC) {
               semblTA *= semblOriginalCC;
               semblTA /= semblCC;
               semblCC = semblOriginalCC;
            }
            else if (semblTA > semblOriginalTA) {
               semblCC *= semblOriginalTA;
               semblCC /= semblTA;
               semblTA = semblOriginalTA;
            }
         }

         resultatSemblancaCicadettaCerdaniensis = new Pair(semblCC,
resultatSemblancaCicadettaCerdaniensis.getSecond());
         resultatSemblancaTettigettalnaArgentata = new Pair(semblTA,
resultatSemblancaTettigettalnaArgentata.getSecond());
      }

      //Cicadetta cerdaniensis / Tettigettula pygmea
      if (semblCC > 0.25 && semblTP > 0.25) {
         double semblOriginalCC = semblCC;
         double semblOriginalTP = semblTP;

         if (duradaMitjanaCantCurt > 0) {
            double coefDuradaMitjanaCant = ((130 - duradaMitjanaCantCurt) / 130) * 0.5
* nivellConfiancaDuradaCantsCurts;
            if (coefDuradaMitjanaCant > 0) {
               semblCC *= 1 + coefDuradaMitjanaCant;
               semblTP /= 1 + coefDuradaMitjanaCant;
            }
            else {
               coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 180) / 180) * 0.75 *
nivellConfiancaDuradaCantsCurts;
               if (coefDuradaMitjanaCant > 0) {
                  semblCC /= 1 + coefDuradaMitjanaCant;
                  semblTP *= 1 + coefDuradaMitjanaCant;
               }
            }
         }

         if (duradaMitjanaSilenci > 0) {
            double coefDuradaMitjanaSilenci = ((170 - duradaMitjanaSilenci) / 170) * 0.5;
            if (coefDuradaMitjanaSilenci > 0) {
               semblCC /= 1 + coefDuradaMitjanaSilenci;
               semblTP *= 1 + coefDuradaMitjanaSilenci;
            }
            else {
               coefDuradaMitjanaSilenci = ((duradaMitjanaSilenci - 200) / 200) * 0.65;
               if (coefDuradaMitjanaSilenci > 0) {
                  semblCC *= 1 + coefDuradaMitjanaSilenci;
                  semblTP /= 1 + coefDuradaMitjanaSilenci;
               }
            }
         }

         if (frequenciaMaximaMostreig >= 16000) {
            double coefMitjFreq = ((15000 - frequenciaMitjana - incrementFreqCC) /
15000) * 1.35;
            if (coefMitjFreq > 0) {
               semblCC *= 1 + coefMitjFreq;
               semblTP /= 1 + coefMitjFreq;
            }
            else {
               coefMitjFreq = ((frequenciaMitjana + incrementFreqCC - 18500) / 18500) *
1.55;
               if (coefMitjFreq > 0) {
                  semblCC /= 1 + coefMitjFreq;
                  semblTP *= 1 + coefMitjFreq;
               }
            }
            coefMitjFreq = ((15000 - frequenciaMitjana - incrementFreqTP) / 15000) *
1.35;
            if (coefMitjFreq > 0) {
               semblCC *= 1 + coefMitjFreq;
               semblTP /= 1 + coefMitjFreq;
            }
            else {
               coefMitjFreq = ((frequenciaMitjana + incrementFreqTP - 18500) / 18500) *
1.55;
               if (coefMitjFreq > 0) {
                  semblCC /= 1 + coefMitjFreq;
                  semblTP *= 1 + coefMitjFreq;
               }
            }
         }

         semblCC /= 1 + nBatecsAlesOCantsIncorrectes * 0.05;
         semblTP *= 1 + nBatecsAlesOCantsIncorrectes * 0.05;

         if (proporcioSilenci > 0) {
            double coefPropSil = (proporcioSilenci - 0.7) * 1.75;
            if (coefPropSil > 0) {
               semblCC *= 1 + coefPropSil;
               semblTP /= 1 + coefPropSil;
            }
            else {
               semblCC /= 1 - coefPropSil;
               semblTP *= 1 - coefPropSil;
            }
         }

         if (semblCC > 1) {
            double varAux = semblCC;
            semblCC /= varAux;
            semblTP /= varAux;
         }
         if (semblTP > 1) {
            double varAux = semblTP;
            semblCC /= varAux;
            semblTP /= varAux;
         }

         semblCC = Math.max(0, Math.min(1, semblCC));
         semblTP = Math.max(0, Math.min(1, semblTP));

         if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblCC > semblOriginalCC) {
               semblTP *= semblOriginalCC;
               semblTP /= semblCC;
               semblCC = semblOriginalCC;
            }
            else if (semblTP > semblOriginalTP) {
               semblCC *= semblOriginalTP;
               semblCC /= semblTP;
               semblTP = semblOriginalTP;
            }
         }

         resultatSemblancaCicadettaCerdaniensis = new Pair(semblCC,
resultatSemblancaCicadettaCerdaniensis.getSecond());
         resultatSemblancaTettigettulaPygmea = new Pair(semblTP,
resultatSemblancaTettigettulaPygmea.getSecond());
      }

      //Cicadetta cerdaniensis / Tibicina corsica fairmairei
      if (semblCC > 0.25 && semblTC > 0.25) {
         double semblOriginalCC = semblCC;
         double semblOriginalTC = semblTC;

         double coefNivConfSonCantsCurts = ((1 - nivellConfiancaSonCantsCurts)) * 0.8;
         semblCC /= 1 + coefNivConfSonCantsCurts;
         semblTC *= 1 + coefNivConfSonCantsCurts;

         double coefNivConfDuradaCantsCurts = ((1 - nivellConfiancaDuradaCantsCurts))
* 0.3;
         semblCC /= 1 + coefNivConfDuradaCantsCurts;
         semblTC *= 1 + coefNivConfDuradaCantsCurts;
```

```
        double coefNivConfNGP = ((nivellConfiancaNombreGrupsDePolsosPerSegon - 0.7)) * 0.8;
        if (coefNivConfNGP > 0) {
          semblCC /= 1 + coefNivConfNGP;
          semblTC *= 1 + coefNivConfNGP;
        }
        else {
          semblCC *= 1 - coefNivConfNGP;
          semblTC /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((59 - nombreGrupsDePolsosPerSegon) / 59) * 3.5 * nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblCC *= 1 + coefNGP;
            semblTC /= 1 + coefNGP;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 65) / 65) * 3.5 * nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblCC *= 1 + coefNGP;
              semblTC /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 59 && nombreGrupsDePolsosPerSegon <= 65) {
              semblCC /= 1.25;
              semblTC *= 1.25;
            }
          }
        }

        if (semblCC > 1) {
          double varAux = semblCC;
          semblCC /= varAux;
          semblTC /= varAux;
        }
        if (semblTC > 1) {
          double varAux = semblTC;
          semblCC /= varAux;
          semblTC /= varAux;
        }

        semblCC = Math.max(0, Math.min(1, semblCC));
        semblTC = Math.max(0, Math.min(1, semblTC));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCC > semblOriginalCC) {
            semblTC *= semblOriginalCC;
            semblTC /= semblCC;
            semblCC = semblOriginalCC;
          }
          else if (semblTC > semblOriginalTC) {
            semblCC *= semblOriginalTC;
            semblCC /= semblTC;
            semblTC = semblOriginalTC;
          }
        }

        resultatSemblancaCicadettaCerdaniensis = new Pair(semblCC, resultatSemblancaCicadettaCerdaniensis.getSecond());
        resultatSemblancaTibicinaCorsicaFairmairei = new Pair(semblTC, resultatSemblancaTibicinaCorsicaFairmairei.getSecond());
      }

      //Cicadetta cerdaniensis / Tibicina haematodes
      if (semblCC > 0.25 && semblTH > 0.25) {
        double semblOriginalCC = semblCC;
        double semblOriginalTH = semblTH;

        double coefNivConfSonCantsCurts = ((1 - nivellConfiancaSonCantsCurts)) * 0.8;
        semblCC /= 1 + coefNivConfSonCantsCurts;
        semblTH *= 1 + coefNivConfSonCantsCurts;

        double coefNivConfDuradaCantsCurts = ((1 - nivellConfiancaDuradaCantsCurts)) * 0.3;
        semblCC /= 1 + coefNivConfDuradaCantsCurts;
        semblTH *= 1 + coefNivConfDuradaCantsCurts;

        double coefNivConfNGP = ((nivellConfiancaNombreGrupsDePolsosPerSegon - 0.7)) * 0.8;
        if (coefNivConfNGP > 0) {
          semblCC /= 1 + coefNivConfNGP;
          semblTH *= 1 + coefNivConfNGP;
        }
        else {
          semblCC *= 1 - coefNivConfNGP;
          semblTH /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((90 - nombreGrupsDePolsosPerSegon) / 90) * 3.0 * nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblCC *= 1 + coefNGP;
            semblTH /= 1 + coefNGP;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 105) / 105) * 3.0 * nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblCC *= 1 + coefNGP;
              semblTH /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 90 && nombreGrupsDePolsosPerSegon <= 105) {
              semblCC /= 1.25;
              semblTH *= 1.25;
            }
          }
        }

        if (semblCC > 1) {
          double varAux = semblCC;
          semblCC /= varAux;
          semblTH /= varAux;
        }
        if (semblTH > 1) {
          double varAux = semblTH;
          semblCC /= varAux;
          semblTH /= varAux;
        }

        semblCC = Math.max(0, Math.min(1, semblCC));
        semblTH = Math.max(0, Math.min(1, semblTH));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblCC > semblOriginalCC) {
            semblTH *= semblOriginalCC;
            semblTH /= semblCC;
            semblCC = semblOriginalCC;
          }
          else if (semblTH > semblOriginalTH) {
            semblCC *= semblOriginalTH;
            semblCC /= semblTH;
            semblTH = semblOriginalTH;
          }
        }

        resultatSemblancaCicadettaCerdaniensis = new Pair(semblCC, resultatSemblancaCicadettaCerdaniensis.getSecond());
        resultatSemblancaTibicinaHaematodes = new Pair(semblTH, resultatSemblancaTibicinaHaematodes.getSecond());
      }

      //Hilaphura varipes / Tettigettula pygmea
      if (semblHV > 0.25 && semblTP > 0.25) {
        double semblOriginalHV = semblHV;
        double semblOriginalTP = semblTP;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCant = ((150 - duradaMitjanaCantCurt) / 150) * 0.35 * nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCant > 0) {
            semblHV /= 1 + coefDuradaMitjanaCant;
            semblTP *= 1 + coefDuradaMitjanaCant;
          }
        }

        semblHV /= 1 + nBatecsAlesOCantsIncorrectes * 0.1;
        semblTP *= 1 + nBatecsAlesOCantsIncorrectes * 0.1;

        if (semblHV > 1) {
          double varAux = semblHV;
          semblHV /= varAux;
          semblTP /= varAux;
        }
        if (semblTP > 1) {
          double varAux = semblTP;
          semblHV /= varAux;
          semblTP /= varAux;
        }

        semblHV = Math.max(0, Math.min(1, semblHV));
        semblTP = Math.max(0, Math.min(1, semblTP));
```

```
        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblCC > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblHV > semblOriginalHV) {
            semblTP *= semblOriginalHV;
            semblTP /= semblHV;
            semblHV = semblOriginalHV;
          }
          else if (semblTP > semblOriginalTP) {
            semblHV *= semblOriginalTP;
            semblHV /= semblTP;
            semblTP = semblOriginalTP;
          }
        }

        resultatSemblancaHilaphuraVaripes = new Pair(semblHV,
resultatSemblancaHilaphuraVaripes.getSecond());
        resultatSemblancaTettigettulaPygmea = new Pair(semblTP,
resultatSemblancaTettigettulaPygmea.getSecond());
      }

      //Hilaphura varipes / Tibicina tomentosa
      if (semblHV > 0.25 && semblTT > 0.25) {
        double semblOriginalHV = semblHV;
        double semblOriginalTT = semblTT;

        if (percentatgeCantsCurts == 100) {
          semblHV *= 1.1;
          semblTT /= 1.1;
        }
        else {
          semblHV /= 1.1;
          semblTT *= 1.1;
        }

        if (frequenciaMaximaMostreig >= 9000) {
          double coefFreq = ((7250 - frequenciaMitjana - incrementFreqHV) / 7250) *
0.4;
          if (coefFreq > 0) {
            semblHV *= 1 + coefFreq;
            semblTT /= 1 + coefFreq;
          }
          else {
            coefFreq = ((frequenciaMitjana + incrementFreqHV - 9000) / 9000) * 0.5;
            if (coefFreq > 0) {
              semblHV /= 1 + coefFreq;
              semblTT *= 1 + coefFreq;
            }
          }
          coefFreq = ((7250 - frequenciaMitjana - incrementFreqTT) / 7250) * 0.4;
          if (coefFreq > 0) {
            semblHV *= 1 + coefFreq;
            semblTT /= 1 + coefFreq;
          }
          else {
            coefFreq = ((frequenciaMitjana + incrementFreqTT - 9000) / 9000) * 0.5;
            if (coefFreq > 0) {
              semblHV /= 1 + coefFreq;
              semblTT *= 1 + coefFreq;
            }
          }
        }

        double coefNivConfNGP = ((0.7 -
nivellConfiancaNombreGrupsDePolsosPerSegon)) * 0.8;
        if (coefNivConfNGP > 0) {
          semblHV *= 1 + coefNivConfNGP;
          semblTT /= 1 + coefNivConfNGP;
        }
        else {
          semblHV /= 1 - coefNivConfNGP;
          semblTT *= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((145 - nombreGrupsDePolsosPerSegon) / 145) * 2.5 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblHV *= 1 + coefNGP;
            semblTT /= 1 + coefNGP;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 180) / 180) * 2.5 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblHV *= 1 + coefNGP;
              semblTT /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 145 &&
nombreGrupsDePolsosPerSegon <= 180) {
              semblHV /= 1.25;
              semblTT *= 1.25;
            }
          }
        }

        double coefPropSil = (proporcioSilenci - 0.3) * 0.3;
        if (coefPropSil > 0) {
          semblHV *= 1 + coefPropSil;
          semblTT /= 1 + coefPropSil;
        }
        else {
          coefPropSil = (0.2 - proporcioSilenci) * 0.5;
          if (coefPropSil > 0) {
            semblHV /= 1 + coefPropSil;
            semblTT *= 1 + coefPropSil;
          }
        }

        if (semblHV > 1) {
          double varAux = semblHV;
          semblHV /= varAux;
          semblTT /= varAux;
        }
        if (semblTT > 1) {
          double varAux = semblTT;
          semblHV /= varAux;
          semblTT /= varAux;
        }

        semblHV = Math.max(0, Math.min(1, semblHV));
        semblTT = Math.max(0, Math.min(1, semblTT));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblCC > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC >
0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25) {
          if (semblHV > semblOriginalHV) {
            semblTT *= semblOriginalHV;
            semblTT /= semblHV;
            semblHV = semblOriginalHV;
          }
          else if (semblTT > semblOriginalTT) {
            semblHV *= semblOriginalTT;
            semblHV /= semblTT;
            semblTT = semblOriginalTT;
          }
        }

        resultatSemblancaHilaphuraVaripes = new Pair(semblHV,
resultatSemblancaHilaphuraVaripes.getSecond());
        resultatSemblancaTibicinaTomentosa = new Pair(semblTT,
resultatSemblancaTibicinaTomentosa.getSecond());
      }

      //Lyristes plebejus / Tettigettalna argentata
      if (semblLP > 0.25 && semblTA > 0.25) {
        double semblOriginalLP = semblLP;
        double semblOriginalTA = semblTA;

        if (duradaMitjanaCantCurt > 0) {
          double coefDuradaMitjanaCant = ((duradaMitjanaCantCurt - 35) / 35) * 0.6 *
nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaCant > 0) {
            semblLP *= 1 + coefDuradaMitjanaCant;
            semblTA /= 1 + coefDuradaMitjanaCant;
          }
          else {
            semblLP /= 1 - coefDuradaMitjanaCant;
            semblTA *= 1 - coefDuradaMitjanaCant;
          }
        }

        if (duradaMitjanaSilenci > 0) {
          double coefDuradaMitjanaSilenci = ((duradaMitjanaSilenci - 55) / 55) * 0.6 *
nivellConfiancaDuradaCantsCurts;
          if (coefDuradaMitjanaSilenci > 0) {
            semblLP /= 1 + coefDuradaMitjanaSilenci;
            semblTA *= 1 + coefDuradaMitjanaSilenci;
          }
          else {
            semblLP *= 1 - coefDuradaMitjanaSilenci;
            semblTA /= 1 - coefDuradaMitjanaSilenci;
          }
        }

        if (frequenciaMaximaMostreig >= 11000) {
          double coefFreq = ((frequenciaMitjana + incrementFreqLP - 9500) / 9500) *
0.3;
          if (coefFreq > 0) {
            semblLP /= 1 + coefFreq;
            semblTA *= 1 + coefFreq;
          }
```

```
            else {
               semblLP *= 1 - coefFreq;
               semblTA /= 1 - coefFreq;
            }
            coefFreq = ((frequenciaMitjana + incrementFreqTA - 9500) / 9500) * 0.3;
            if (coefFreq > 0) {
               semblLP /= 1 + coefFreq;
               semblTA *= 1 + coefFreq;
            }
            else {
               semblLP *= 1 - coefFreq;
               semblTA /= 1 - coefFreq;
            }
         }

         semblLP *= 1 + ((100 - percentatgeCantsCurts) / 100) * 5;
         semblTA /= 1 + ((100 - percentatgeCantsCurts) / 100) * 5;

         if (semblLP > 1) {
            double varAux = semblLP;
            semblLP /= varAux;
            semblTA /= varAux;
         }
         if (semblTA > 1) {
            double varAux = semblTA;
            semblLP /= varAux;
            semblTA /= varAux;
         }

         semblLP = Math.max(0, Math.min(1, semblLP));
         semblTA = Math.max(0, Math.min(1, semblTA));

         if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblCC > 0.25 || semblHV > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblLP > semblOriginalLP) {
               semblTA *= semblOriginalLP;
               semblTA /= semblLP;
               semblLP = semblOriginalLP;
            }
            else if (semblTA > semblOriginalTA) {
               semblLP *= semblOriginalTA;
               semblLP /= semblTA;
               semblTA = semblOriginalTA;
            }
         }

         resultatSemblancaLyristesPlebejus = new Pair(semblLP,
resultatSemblancaLyristesPlebejus.getSecond());
         resultatSemblancaTettigettalnaArgentata = new Pair(semblTA,
resultatSemblancaTettigettalnaArgentata.getSecond());
      }

      //Lyristes plebejus / Tettigettula pygmea
      if (semblLP > 0.25 && semblTP > 0.25) {
         double semblOriginalLP = semblLP;
         double semblOriginalTP = semblTP;

         if (98 > duradaMitjanaCantCurt && duradaMitjanaCantCurt > 0) semblTP *= 0;

         if (semblLP > 1) {
            double varAux = semblLP;
            semblLP /= varAux;
            semblTP /= varAux;
         }
         if (semblTP > 1) {
            double varAux = semblTP;
            semblLP /= varAux;
            semblTP /= varAux;
         }

         semblLP = Math.max(0, Math.min(1, semblLP));
         semblTP = Math.max(0, Math.min(1, semblTP));

         if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblCC > 0.25 || semblHV > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblLP > semblOriginalLP) {
               semblTP *= semblOriginalLP;
               semblTP /= semblLP;
               semblLP = semblOriginalLP;
            }
            else if (semblTP > semblOriginalTP) {
               semblLP *= semblOriginalTP;
               semblLP /= semblTP;
               semblTP = semblOriginalTP;
            }
         }

         resultatSemblancaLyristesPlebejus = new Pair(semblLP,
resultatSemblancaLyristesPlebejus.getSecond());
         resultatSemblancaTettigettulaPygmea = new Pair(semblTP,
resultatSemblancaTettigettulaPygmea.getSecond());
      }

      //Lyristes plebejus / Tibicina corsica fairmairei
      if (semblLP > 0.25 && semblTC > 0.25) {
         double semblOriginalLP = semblLP;
         double semblOriginalTC = semblTC;

         if (percentatgeCantsCurts > 0) {
            semblLP *= 1.1;
            semblTC /= 1.1;
         }
         else {
            semblLP /= 1.1;
            semblTC *= 1.1;
         }

         if (frequenciaMaximaMostreig >= 9000) {
            double coefFreq = ((frequenciaMitjana + incrementFreqLP - 9000) / 9000) *
1.2;
            if (coefFreq > 0) {
               semblLP /= 1 + coefFreq;
               semblTC *= 1 + coefFreq;
            }
            else {
               semblLP *= 1 - coefFreq;
               semblTC /= 1 - coefFreq;
            }
            coefFreq = ((frequenciaMitjana + incrementFreqTC - 9000) / 9000) * 1.2;
            if (coefFreq > 0) {
               semblLP /= 1 + coefFreq;
               semblTC *= 1 + coefFreq;
            }
            else {
               semblLP *= 1 - coefFreq;
               semblTC /= 1 - coefFreq;
            }
         }

         double coefNivConfNGP = ((0.7 -
nivellConfiancaNombreGrupsDePolsosPerSegon)) * 0.6;
         if (coefNivConfNGP > 0) {
            semblLP *= 1 + coefNivConfNGP;
            semblTC /= 1 + coefNivConfNGP;
         }
         else {
            semblLP /= 1 - coefNivConfNGP;
            semblTC *= 1 - coefNivConfNGP;
         }

         if (nombreGrupsDePolsosPerSegon > 0) {
            double coefNGP = ((59 - nombreGrupsDePolsosPerSegon) / 59) * 3.5 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
               semblLP *= 1 + coefNGP;
               semblTC /= 1 + coefNGP;
            }
            else {
               coefNGP = ((nombreGrupsDePolsosPerSegon - 65) / 65) * 3.5 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
               if (coefNGP > 0) {
                  semblLP *= 1 + coefNGP;
                  semblTC /= 1 + coefNGP;
               }
               else if (nombreGrupsDePolsosPerSegon >= 59 &&
nombreGrupsDePolsosPerSegon <= 65) {
                  semblLP /= 1.25;
                  semblTC *= 1.25;
               }
            }
         }

         if (semblLP > 1) {
            double varAux = semblLP;
            semblLP /= varAux;
            semblTC /= varAux;
         }
         if (semblTC > 1) {
            double varAux = semblTC;
            semblLP /= varAux;
            semblTC /= varAux;
         }

         semblLP = Math.max(0, Math.min(1, semblLP));
         semblTC = Math.max(0, Math.min(1, semblTC));

         if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblCC > 0.25 || semblHV > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblLP > semblOriginalLP) {
```

```
            semblTC *= semblOriginalLP;
            semblTC /= semblLP;
            semblLP = semblOriginalLP;
          }
          else if (semblTC > semblOriginalTC) {
            semblLP *= semblOriginalTC;
            semblLP /= semblTC;
            semblTC = semblOriginalTC;
          }
        }

        resultatSemblancaLyristesPlebejus = new Pair(semblLP,
resultatSemblancaLyristesPlebejus.getSecond());
        resultatSemblancaTibicinaCorsicaFairmairei = new Pair(semblTC,
resultatSemblancaTibicinaCorsicaFairmairei.getSecond());
      }

      //Lyristes plebejus / Tibicina garricola
      if (semblLP > 0.25 && semblTG > 0.25) {
        double semblOriginalLP = semblLP;
        double semblOriginalTG = semblTG;

        if (percentatgeCantsCurts > 0) {
          semblLP *= 1.1;
          semblTG /= 1.1;
        }
        else {
          semblLP /= 1.1;
          semblTG *= 1.1;
        }

        if (frequenciaMaximaMostreig >= 9000) {
          double coefFreq = ((7500 - frequenciaMitjana - incrementFreqLP) / 7750) *
0.85;
          if (coefFreq > 0) {
            semblLP *= 1 + coefFreq;
            semblTG /= 1 + coefFreq;
          }
          else {
            coefFreq = ((frequenciaMitjana + incrementFreqLP - 9000) / 9000) * 0.95;
            if (coefFreq > 0) {
              semblLP /= 1 + coefFreq;
              semblTG *= 1 + coefFreq;
            }
          }
          coefFreq = ((7500 - frequenciaMitjana - incrementFreqTG) / 7750) * 0.85;
          if (coefFreq > 0) {
            semblLP *= 1 + coefFreq;
            semblTG /= 1 + coefFreq;
          }
          else {
            coefFreq = ((frequenciaMitjana + incrementFreqTG - 9000) / 9000) * 0.95;
            if (coefFreq > 0) {
              semblLP /= 1 + coefFreq;
              semblTG *= 1 + coefFreq;
            }
          }
        }

        double coefNivConfNGP = ((0.7 -
nivellConfiancaNombreGrupsDePolsosPerSegon)) * 0.5;
        if (coefNivConfNGP > 0) {
          semblLP *= 1 + coefNivConfNGP;
          semblTG /= 1 + coefNivConfNGP;
        }
        else {
          semblLP /= 1 - coefNivConfNGP;
          semblTG *= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((66 - nombreGrupsDePolsosPerSegon) / 66) * 3.25 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblLP *= 1 + coefNGP;
            semblTG /= 1 + coefNGP;
          }
          else {
            if (nombreGrupsDePolsosPerSegon > 132 &&
nombreGrupsDePolsosPerSegon < 142) {
              semblLP *= 1.2;
              semblTG /= 1.2;
            }
            else {
              coefNGP = ((nombreGrupsDePolsosPerSegon - 71) / 71) * 3.25 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
              if (coefNGP > 0) {
                semblLP *= 1 + coefNGP;
                semblTG /= 1 + coefNGP;
              }
              else if (nombreGrupsDePolsosPerSegon >= 66 &&
nombreGrupsDePolsosPerSegon <= 71) {
                semblLP /= 1.25;
                semblTG *= 1.25;
              }
            }
          }
        }

        if (semblLP > 1) {
          double varAux = semblLP;
          semblLP /= varAux;
          semblTG /= varAux;
        }
        if (semblTG > 1) {
          double varAux = semblTG;
          semblLP /= varAux;
          semblTG /= varAux;
        }

        semblLP = Math.max(0, Math.min(1, semblLP));
        semblTG = Math.max(0, Math.min(1, semblTG));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblCC > 0.25 || semblHV > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblLP > semblOriginalLP) {
            semblTG *= semblOriginalLP;
            semblTG /= semblLP;
            semblLP = semblOriginalLP;
          }
          else if (semblTG > semblOriginalTG) {
            semblLP *= semblOriginalTG;
            semblLP /= semblTG;
            semblTG = semblOriginalTG;
          }
        }

        resultatSemblancaLyristesPlebejus = new Pair(semblLP,
resultatSemblancaLyristesPlebejus.getSecond());
        resultatSemblancaTibicinaGarricola = new Pair(semblTG,
resultatSemblancaTibicinaGarricola.getSecond());
      }

      //Lyristes plebejus / Tibicina haematodes
      if (semblLP > 0.25 && semblTH > 0.25) {
        double semblOriginalLP = semblLP;
        double semblOriginalTH = semblTH;

        if (percentatgeCantsCurts > 0) {
          semblLP *= 1.1;
          semblTH /= 1.1;
        }
        else {
          semblLP /= 1.1;
          semblTH *= 1.1;
        }

        double coefNivConfNGP = ((nivellConfiancaNombreGrupsDePolsosPerSegon -
0.75)) * 0.7;
        if (coefNivConfNGP > 0) {
          semblLP /= 1 + coefNivConfNGP;
          semblTH *= 1 + coefNivConfNGP;
        }
        else {
          semblLP *= 1 - coefNivConfNGP;
          semblTH /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((90 - nombreGrupsDePolsosPerSegon) / 90) * 3.0 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblLP *= 1 + coefNGP;
            semblTH /= 1 + coefNGP;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 105) / 105) * 3.0 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblLP *= 1 + coefNGP;
              semblTH /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 90 &&
nombreGrupsDePolsosPerSegon <= 105) {
              semblLP /= 1.25;
              semblTH *= 1.25;
            }
          }
        }
```

```
        if (semblLP > 1) {
          double varAux = semblLP;
          semblLP /= varAux;
          semblTH /= varAux;
        }
        if (semblTH > 1) {
          double varAux = semblTH;
          semblLP /= varAux;
          semblTH /= varAux;
        }

        semblLP = Math.max(0, Math.min(1, semblLP));
        semblTH = Math.max(0, Math.min(1, semblTH));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblLP > semblOriginalLP) {
            semblTH *= semblOriginalLP;
            semblTH /= semblLP;
            semblLP = semblOriginalLP;
          }
          else if (semblTH > semblOriginalTH) {
            semblLP *= semblOriginalTH;
            semblLP /= semblTH;
            semblTH = semblOriginalTH;
          }
        }

        resultatSemblancaLyristesPlebejus = new Pair(semblLP, resultatSemblancaLyristesPlebejus.getSecond());
        resultatSemblancaTibicinaHaematodes = new Pair(semblTH, resultatSemblancaTibicinaHaematodes.getSecond());
      }

      //Lyristes plebejus / Tibicina quadrisignata
      if (semblLP > 0.25 && semblTQ > 0.25) {
        double semblOriginalLP = semblLP;
        double semblOriginalTQ = semblTQ;

        if (duradaMitjanaCantCurt > 0) {
          double coefDurMitjCantCurt = ((duradaMitjanaCantCurt - 100) / 100) * 0.4 * nivellConfiancaDuradaCantsCurts;
          if (coefDurMitjCantCurt > 0) {
            semblLP /= 1 + coefDurMitjCantCurt;
            semblTQ *= 1 + coefDurMitjCantCurt;
          }
          else {
            semblLP *= 1 - coefDurMitjCantCurt;
            semblTQ /= 1 - coefDurMitjCantCurt;
          }
        }

        semblLP /= 1 + Math.max(0, (pendentPrimeraMeitatCantsLlargs - 1.2) * 0.4);
        semblTQ *= 1 + Math.max(0, (pendentPrimeraMeitatCantsLlargs - 1.2) * 0.4);

        semblLP /= 1 + Math.max(0, (pendentSegonaMeitatCantsLlargs - 1.2) * 0.4);
        semblTQ *= 1 + Math.max(0, (pendentSegonaMeitatCantsLlargs - 1.2) * 0.4);

        double coefNivConfNGP = ((nivellConfiancaNombreGrupsDePolsosPerSegon - 0.7)) * 0.6;
        if (coefNivConfNGP > 0) {
          semblLP /= 1 + coefNivConfNGP;
          semblTQ *= 1 + coefNivConfNGP;
        }
        else {
          semblLP *= 1 - coefNivConfNGP;
          semblTQ /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = ((73 - nombreGrupsDePolsosPerSegon) / 73) * 3.0 * nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblLP *= 1 + coefNGP;
            semblTQ /= 1 + coefNGP;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 80) / 80) * 3.0 * nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblLP *= 1 + coefNGP;
              semblTQ /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 73 && nombreGrupsDePolsosPerSegon <= 80) {
              semblLP /= 1.25;
              semblTQ *= 1.25;
            }
          }
        }

        if (frequenciaMaximaMostreig >= 9000) {
          double coefFreq = ((7500 - frequenciaMitjana - incrementFreqLP) / 7500) * 0.45;
          if (coefFreq > 0) {
            semblLP *= 1 + coefFreq;
            semblTQ /= 1 + coefFreq;
          }
          else {
            coefFreq = ((frequenciaMitjana + incrementFreqLP - 9250) / 9250) * 0.55;
            if (coefFreq > 0) {
              semblLP /= 1 + coefFreq;
              semblTQ *= 1 + coefFreq;
            }
          }
          coefFreq = ((7500 - frequenciaMitjana - incrementFreqTQ) / 7500) * 0.45;
          if (coefFreq > 0) {
            semblLP *= 1 + coefFreq;
            semblTQ /= 1 + coefFreq;
          }
          else {
            coefFreq = ((frequenciaMitjana + incrementFreqTQ - 9250) / 9250) * 0.55;
            if (coefFreq > 0) {
              semblLP /= 1 + coefFreq;
              semblTQ *= 1 + coefFreq;
            }
          }
        }

        semblLP /= 1 + Math.max(0, (proporcioSilenci - 0.4) * 0.15);
        semblTQ *= 1 + Math.max(0, (proporcioSilenci - 0.4) * 0.15);

        if (semblLP > 1) {
          double varAux = semblLP;
          semblLP /= varAux;
          semblTQ /= varAux;
        }
        if (semblTQ > 1) {
          double varAux = semblTQ;
          semblLP /= varAux;
          semblTQ /= varAux;
        }

        semblLP = Math.max(0, Math.min(1, semblLP));
        semblTQ = Math.max(0, Math.min(1, semblTQ));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTT > 0.25) {
          if (semblLP > semblOriginalLP) {
            semblTQ *= semblOriginalLP;
            semblTQ /= semblLP;
            semblLP = semblOriginalLP;
          }
          else if (semblTQ > semblOriginalTQ) {
            semblLP *= semblOriginalTQ;
            semblLP /= semblTQ;
            semblTQ = semblOriginalTQ;
          }
        }

        resultatSemblancaLyristesPlebejus = new Pair(semblLP, resultatSemblancaLyristesPlebejus.getSecond());
        resultatSemblancaTibicinaQuadrisignata = new Pair(semblTQ, resultatSemblancaTibicinaQuadrisignata.getSecond());
      }

      //Lyristes plebejus / Tibicina tomentosa
      if (semblLP > 0.25 && semblTT > 0.25) {
        double semblOriginalLP = semblLP;
        double semblOriginalTT = semblTT;

        if (percentatgeCantsCurts == 100) {
          semblLP *= 1.2;
          semblTT /= 1.2;
        }
        else if (percentatgeCantsCurts == 0) {
          semblLP /= 1.1;
          semblTT *= 1.1;
        }
        else {
          semblLP *= 1.1;
          semblTT /= 1.1;
        }

        if (frequenciaMaximaMostreig >= 9000) {
          double coefFreq = ((7750 - frequenciaMitjana - incrementFreqLP) / 7750) * 0.6;
          if (coefFreq > 0) {
            semblLP *= 1 + coefFreq;
            semblTT /= 1 + coefFreq;
```

```
            }
            else {
                coefFreq = ((frequenciaMitjana + incrementFreqLP - 8750) / 8750) * 0.7;
                if (coefFreq > 0) {
                    semblLP /= 1 + coefFreq;
                    semblTT *= 1 + coefFreq;
                }
            }
            coefFreq = ((7750 - frequenciaMitjana - incrementFreqTT) / 7750) * 0.6;
            if (coefFreq > 0) {
                semblLP *= 1 + coefFreq;
                semblTT /= 1 + coefFreq;
            }
            else {
                coefFreq = ((frequenciaMitjana + incrementFreqTT - 8750) / 8750) * 0.7;
                if (coefFreq > 0) {
                    semblLP /= 1 + coefFreq;
                    semblTT *= 1 + coefFreq;
                }
            }
        }

        double coefNivConfNGP = ((0.8 - nivellConfiancaNombreGrupsDePolsosPerSegon)) * 0.7;
        if (coefNivConfNGP > 0) {
            semblLP *= 1 + coefNivConfNGP;
            semblTT /= 1 + coefNivConfNGP;
        }
        else {
            semblLP /= 1 - coefNivConfNGP;
            semblTT *= 1 - coefNivConfNGP;
        }

        double coefDefNGP = ((0.55 - definicioGrupsDePolsos)) * 0.4;
        if (coefDefNGP > 0) {
            semblLP *= 1 + coefDefNGP;
            semblTT /= 1 + coefDefNGP;
        }
        else {
            semblLP /= 1 - coefDefNGP;
            semblTT *= 1 - coefDefNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
            double coefNGP = ((145 - nombreGrupsDePolsosPerSegon) / 145) * 2.5 * nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
                semblLP *= 1 + coefNGP;
                semblTT /= 1 + coefNGP;
            }
            else {
                coefNGP = ((nombreGrupsDePolsosPerSegon - 180) / 180) * 2.5 * nivellConfiancaNombreGrupsDePolsosPerSegon;
                if (coefNGP > 0) {
                    semblLP *= 1 + coefNGP;
                    semblTT /= 1 + coefNGP;
                }
                else if (nombreGrupsDePolsosPerSegon >= 145 && nombreGrupsDePolsosPerSegon <= 180) {
                    semblLP /= 1.25;
                    semblTT *= 1.25;
                }
            }
        }

        double coefCantsLP = nivellConfiancaCantsLP * 7.5;
        semblLP *= 1 + coefCantsLP;
        semblTT /= 1 + coefCantsLP;

        if (semblLP > 1) {
            double varAux = semblLP;
            semblLP /= varAux;
            semblTT /= varAux;
        }
        if (semblTT > 1) {
            double varAux = semblTT;
            semblLP /= varAux;
            semblTT /= varAux;
        }

        semblLP = Math.max(0, Math.min(1, semblLP));
        semblTT = Math.max(0, Math.min(1, semblTT));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25) {
            if (semblLP > semblOriginalLP) {
                semblTT *= semblOriginalLP;
                semblTT /= semblLP;
                semblLP = semblOriginalLP;
            }
            else if (semblTT > semblOriginalTT) {
                semblLP *= semblOriginalTT;
                semblLP /= semblTT;
                semblTT = semblOriginalTT;
            }
        }

        resultatSemblancaLyristesPlebejus = new Pair(semblLP, resultatSemblancaLyristesPlebejus.getSecond());
        resultatSemblancaTibicinaTomentosa = new Pair(semblTT, resultatSemblancaTibicinaTomentosa.getSecond());
    }

    //Tettigettalna argentata / Tibicina corsica fairmairei
    if (semblTA > 0.25 && semblTC > 0.25) {
        double semblOriginalTA = semblTA;
        double semblOriginalTC = semblTC;

        if (percentatgeCantsCurts > 0) {
            semblTA *= 1.1;
            semblTC /= 1.1;
        }
        else {
            semblTA /= 1.1;
            semblTC *= 1.1;
        }


        semblTA /= 1 + (1 - nivellConfiancaSonCantsCurts) * 0.8;
        semblTC *= 1 + (1 - nivellConfiancaSonCantsCurts) * 0.8;

        semblTA /= 1 + (1 - nivellConfiancaDuradaCantsCurts) * 0.5;
        semblTC *= 1 + (1 - nivellConfiancaDuradaCantsCurts) * 0.5;


        if (frequenciaMaximaMostreig >= 9000) {
            double coefFreq = ((frequenciaMitjana + incrementFreqTA - 9000) / 9000) * 0.85;
            if (coefFreq > 0) {
                semblTA /= 1 + coefFreq;
                semblTC *= 1 + coefFreq;
            }
            else {
                coefFreq = ((frequenciaMitjana + incrementFreqTA - 11000) / 11000) * 1.05;
                if (coefFreq > 0) {
                    semblTA *= 1 + coefFreq;
                    semblTC /= 1 + coefFreq;
                }
            }
            coefFreq = ((frequenciaMitjana + incrementFreqTC - 9000) / 9000) * 0.85;
            if (coefFreq > 0) {
                semblTA /= 1 + coefFreq;
                semblTC *= 1 + coefFreq;
            }
            else {
                coefFreq = ((frequenciaMitjana + incrementFreqTC - 11000) / 11000) * 1.05;
                if (coefFreq > 0) {
                    semblTA *= 1 + coefFreq;
                    semblTC /= 1 + coefFreq;
                }
            }
        }

        double coefNivConfNGP = (0.75 - nivellConfiancaNombreGrupsDePolsosPerSegon) * 0.8;
        if (coefNivConfNGP > 0) {
            semblTA *= 1 + coefNivConfNGP;
            semblTC /= 1 + coefNivConfNGP;
        }
        else {
            semblTA /= 1 - coefNivConfNGP;
            semblTC *= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
            double coefNGP = ((59 - nombreGrupsDePolsosPerSegon) / 59) * 3.5 * nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
                semblTA *= 1 + coefNGP;
                semblTC /= 1 + coefNGP;
            }
            else {
                coefNGP = ((nombreGrupsDePolsosPerSegon - 65) / 65) * 3.5 * nivellConfiancaNombreGrupsDePolsosPerSegon;
                if (coefNGP > 0) {
                    semblTA *= 1 + coefNGP;
                    semblTC /= 1 + coefNGP;
                }
                else if (nombreGrupsDePolsosPerSegon >= 59 && nombreGrupsDePolsosPerSegon <= 65) {
                    semblTA /= 1.25;
```

```
            semblTC *= 1.25;
          }
        }
      }

      double coefPropSil = (proporcioSilenci - 0.7) * 0.75;
      if (coefPropSil > 0) {
        semblTA *= 1 + coefPropSil;
        semblTC /= 1 + coefPropSil;
      }
      else {
        semblTA /= 1 - coefPropSil;
        semblTC *= 1 - coefPropSil;
      }

      if (semblTA > 1) {
        double varAux = semblTA;
        semblTA /= varAux;
        semblTC /= varAux;
      }
      if (semblTC > 1) {
        double varAux = semblTC;
        semblTA /= varAux;
        semblTC /= varAux;
      }

      semblTA = Math.max(0, Math.min(1, semblTA));
      semblTC = Math.max(0, Math.min(1, semblTC));

      if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblCC > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTP > 0.25 || semblTG >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
        if (semblTA > semblOriginalTA) {
          semblTC *= semblOriginalTA;
          semblTC /= semblTA;
          semblTA = semblOriginalTA;
        }
        else if (semblTC > semblOriginalTC) {
          semblTA *= semblOriginalTC;
          semblTA /= semblTC;
          semblTC = semblOriginalTC;
        }
      }

      resultatSemblancaTettigettalnaArgentata = new Pair(semblTA,
resultatSemblancaTettigettalnaArgentata.getSecond());
      resultatSemblancaTibicinaCorsicaFairmairei = new Pair(semblTC,
resultatSemblancaTibicinaCorsicaFairmairei.getSecond());
    }

    //Tettigettula pygmea / Tibicina garricola
    if (semblTP > 0.25 && semblTG > 0.25) {
      double semblOriginalTP = semblTP;
      double semblOriginalTG = semblTG;

      if (percentatgeCantsCurts > 0) {
        semblTP *= 1.1;
        semblTG /= 1.1;
      }
      else {
        semblTP /= 1.1;
        semblTG *= 1.1;
      }

      double coefNivConfSonCantsCurts = (nivellConfiancaSonCantsCurts - 0.9) * 1.25;
      if (coefNivConfSonCantsCurts > 0) {
        semblTP *= 1 + coefNivConfSonCantsCurts;
        semblTG /= 1 + coefNivConfSonCantsCurts;
      }
      else {
        semblTP /= 1 - coefNivConfSonCantsCurts;
        semblTG *= 1 - coefNivConfSonCantsCurts;
      }

      double coefNivConfNGP = (0.7 -
nivellConfiancaNombreGrupsDePolsosPerSegon) * 0.8;
      if (coefNivConfNGP > 0) {
        semblTP *= 1 + coefNivConfNGP;
        semblTG /= 1 + coefNivConfNGP;
      }
      else {
        semblTP /= 1 - coefNivConfNGP;
        semblTG *= 1 - coefNivConfNGP;
      }

      if (nombreGrupsDePolsosPerSegon > 0) {
        double coefNGP = ((66 - nombreGrupsDePolsosPerSegon) / 66) * 3.25 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
        if (coefNGP > 0) {
          semblTP *= 1 + coefNGP;
          semblTG /= 1 + coefNGP;
        }
        else {
          if (nombreGrupsDePolsosPerSegon > 132 &&
nombreGrupsDePolsosPerSegon < 142) {
            semblTP *= 1.2;
            semblTG /= 1.2;
          }
          else {
            coefNGP = ((nombreGrupsDePolsosPerSegon - 71) / 71) * 3.25 *
nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
              semblTP *= 1 + coefNGP;
              semblTG /= 1 + coefNGP;
            }
            else if (nombreGrupsDePolsosPerSegon >= 66 &&
nombreGrupsDePolsosPerSegon <= 71) {
              semblTP /= 1.25;
              semblTG *= 1.25;
            }
          }
        }
      }

      semblTP *= 1 + nBatecsAlesOCantsIncorrectes * 0.05;
      semblTG /= 1 + nBatecsAlesOCantsIncorrectes * 0.05;

      if (semblTP > 1) {
        double varAux = semblTP;
        semblTP /= varAux;
        semblTG /= varAux;
      }
      if (semblTG > 1) {
        double varAux = semblTG;
        semblTP /= varAux;
        semblTG /= varAux;
      }

      semblTP = Math.max(0, Math.min(1, semblTP));
      semblTG = Math.max(0, Math.min(1, semblTG));

      if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 ||
semblCC > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC >
0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
        if (semblTP > semblOriginalTP) {
          semblTG *= semblOriginalTP;
          semblTG /= semblTP;
          semblTP = semblOriginalTP;
        }
        else if (semblTG > semblOriginalTG) {
          semblTP *= semblOriginalTG;
          semblTP /= semblTG;
          semblTG = semblOriginalTG;
        }
      }

      resultatSemblancaTettigettulaPygmea = new Pair(semblTP,
resultatSemblancaTettigettulaPygmea.getSecond());
      resultatSemblancaTibicinaGarricola = new Pair(semblTG,
resultatSemblancaTibicinaGarricola.getSecond());
    }

    //Tettigettula pygmea / Tibicina haematodes
    if (semblTP > 0.25 && semblTH > 0.25) {
      double semblOriginalTP = semblTP;
      double semblOriginalTH = semblTH;

      double coefPercCantsCurts = ((percentatgeCantsCurts - 95) / 95) * 1.25;
      if (coefPercCantsCurts > 0) {
        semblTP *= 1 + coefPercCantsCurts;
        semblTH /= 1 + coefPercCantsCurts;
      }
      else {
        semblTP /= 1 - coefPercCantsCurts;
        semblTH *= 1 - coefPercCantsCurts;
      }

      double coefNivConfSonCantsCurts = (nivellConfiancaSonCantsCurts - 0.95) *
0.75;
      if (coefNivConfSonCantsCurts > 0) {
        semblTP *= 1 + coefNivConfSonCantsCurts;
        semblTH /= 1 + coefNivConfSonCantsCurts;
      }
      else {
        semblTP /= 1 - coefNivConfSonCantsCurts;
        semblTH *= 1 - coefNivConfSonCantsCurts;
      }

      double coefNivConfNGP = (nivellConfiancaNombreGrupsDePolsosPerSegon -
0.7) * 0.8;
      if (coefNivConfNGP > 0) {
        semblTP /= 1 + coefNivConfNGP;
```

```
            semblTH *= 1 + coefNivConfNGP;
        }
        else {
            semblTP *= 1 - coefNivConfNGP;
            semblTH /= 1 - coefNivConfNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
            double coefNGP = ((90 - nombreGrupsDePolsosPerSegon) / 90) * 3.0 * nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
                semblTP *= 1 + coefNGP;
                semblTH /= 1 + coefNGP;
            }
            else {
                coefNGP = ((nombreGrupsDePolsosPerSegon - 105) / 105) * 3.0 * nivellConfiancaNombreGrupsDePolsosPerSegon;
                if (coefNGP > 0) {
                    semblTP *= 1 + coefNGP;
                    semblTH /= 1 + coefNGP;
                }
                else if (nombreGrupsDePolsosPerSegon >= 90 && nombreGrupsDePolsosPerSegon <= 105) {
                    semblTP /= 1.25;
                    semblTH *= 1.25;
                }
            }
        }

        semblTP *= 1 + nBatecsAlesOCantsIncorrectes * 0.05;
        semblTH /= 1 + nBatecsAlesOCantsIncorrectes * 0.05;

        if (semblTP > 1) {
            double varAux = semblTP;
            semblTP /= varAux;
            semblTH /= varAux;
        }
        if (semblTH > 1) {
            double varAux = semblTH;
            semblTP /= varAux;
            semblTH /= varAux;
        }

        semblTP = Math.max(0, Math.min(1, semblTP));
        semblTH = Math.max(0, Math.min(1, semblTH));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
            if (semblTP > semblOriginalTP) {
                semblTH *= semblOriginalTP;
                semblTH /= semblTP;
                semblTP = semblOriginalTP;
            }
            else if (semblTH > semblOriginalTH) {
                semblTP *= semblOriginalTH;
                semblTP /= semblTH;
                semblTH = semblOriginalTH;
            }
        }

        resultatSemblancaTettigettulaPygmea = new Pair(semblTP, resultatSemblancaTettigettulaPygmea.getSecond());
        resultatSemblancaTibicinaHaematodes = new Pair(semblTH, resultatSemblancaTibicinaHaematodes.getSecond());
    }

    //Tettigettula pygmea / Tibicina tomentosa
    if (semblTP > 0.25 && semblTT > 0.25) {
        double semblOriginalTP = semblTP;
        double semblOriginalTT = semblTT;

        if (duradaMitjanaCantCurt > 160 && duradaMitjanaCantCurt < 310) {
            semblTP *= 1 + 0.5 * nivellConfiancaDuradaCantsCurts * nivellConfiancaDuradaCantsCurts;
            semblTT /= 1 + 0.5 * nivellConfiancaDuradaCantsCurts * nivellConfiancaDuradaCantsCurts;
        }
        else {
            semblTP /= 1 + 0.35 * Math.min(1, Math.min(Math.abs(duradaMitjanaCantCurt - 150), Math.abs(duradaMitjanaCantCurt - 320)) / 50);
            semblTT *= 1 + 0.35 * Math.min(1, Math.min(Math.abs(duradaMitjanaCantCurt - 150), Math.abs(duradaMitjanaCantCurt - 320)) / 50);
        }

        double coefPercCantsCurts = ((percentatgeCantsCurts - 90) / 100) * 0.7;
        if (coefPercCantsCurts > 0) {
            semblTP *= 1 + coefPercCantsCurts;
            semblTT /= 1 + coefPercCantsCurts;
        }
        else {
            semblTP /= 1 - coefPercCantsCurts * 3;
            semblTT *= 1 - coefPercCantsCurts * 3;
        }

        double coefNivConfSonCantsCurts = (nivellConfiancaSonCantsCurts - 0.95) * 0.85;
        if (coefNivConfSonCantsCurts > 0) {
            semblTP *= 1 + coefNivConfSonCantsCurts * 3;
            semblTT /= 1 + coefNivConfSonCantsCurts * 3;
        }
        else {
            semblTP /= 1 - coefNivConfSonCantsCurts;
            semblTT *= 1 - coefNivConfSonCantsCurts;
        }

        double coefNivConfNGP = (0.7 - nivellConfiancaNombreGrupsDePolsosPerSegon) * 0.7;
        if (coefNivConfNGP > 0) {
            semblTP *= 1 + coefNivConfNGP;
            semblTT /= 1 + coefNivConfNGP;
        }
        else {
            semblTP /= 1 - coefNivConfNGP;
            semblTT *= 1 - coefNivConfNGP;
        }

        double coefDefNGP = ((0.4 - definicioGrupsDePolsos)) * 0.4;
        if (coefDefNGP > 0) {
            semblTP *= 1 + coefDefNGP;
            semblTT /= 1 + coefDefNGP;
        }
        else {
            semblTP /= 1 - coefDefNGP;
            semblTT *= 1 - coefDefNGP;
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
            double coefNGP = ((145 - nombreGrupsDePolsosPerSegon) / 145) * 2.5 * nivellConfiancaNombreGrupsDePolsosPerSegon;
            if (coefNGP > 0) {
                semblTP *= 1 + coefNGP;
                semblTT /= 1 + coefNGP;
            }
            else {
                coefNGP = ((nombreGrupsDePolsosPerSegon - 180) / 180) * 2.5 * nivellConfiancaNombreGrupsDePolsosPerSegon;
                if (coefNGP > 0) {
                    semblTP *= 1 + coefNGP;
                    semblTT /= 1 + coefNGP;
                }
                else if (nombreGrupsDePolsosPerSegon >= 145 && nombreGrupsDePolsosPerSegon <= 180) {
                    semblTP /= 1.25;
                    semblTT *= 1.25;
                }
            }
        }

        semblTP *= 1 + nBatecsAlesOCantsIncorrectes * 0.05;
        semblTT /= 1 + nBatecsAlesOCantsIncorrectes * 0.05;

        if (semblTP > 1) {
            double varAux = semblTP;
            semblTP /= varAux;
            semblTT /= varAux;
        }
        if (semblTT > 1) {
            double varAux = semblTT;
            semblTP /= varAux;
            semblTT /= varAux;
        }

        semblTP = Math.max(0, Math.min(1, semblTP));
        semblTT = Math.max(0, Math.min(1, semblTT));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTC > 0.25 || semblTG > 0.25 || semblTH > 0.25 || semblTQ > 0.25) {
            if (semblTP > semblOriginalTP) {
                semblTT *= semblOriginalTP;
                semblTT /= semblTP;
                semblTP = semblOriginalTP;
            }
            else if (semblTT > semblOriginalTT) {
                semblTP *= semblOriginalTT;
                semblTP /= semblTT;
                semblTT = semblOriginalTT;
            }
        }
```

```
        resultatSemblancaTettigettulaPygmea = new Pair(semblTP, resultatSemblancaTettigettulaPygmea.getSecond());
        resultatSemblancaTibicinaTomentosa = new Pair(semblTT, resultatSemblancaTibicinaTomentosa.getSecond());
      }

      //Tibicina corsica fairmairei / Tibicina garricola
      if (semblTC > 0.25 && semblTG > 0.25) {
        double semblOriginalTC = semblTC;
        double semblOriginalTG = semblTG;

        semblTC *= 1 + Math.max(0, (pendentSegonaMeitatCantsLlargs - 1.4) * 0.4);
        semblTG /= 1 + Math.max(0, (pendentSegonaMeitatCantsLlargs - 1.4) * 0.4);

        if (frequenciaMaximaMostreig >= 9000) {
          double coefFreq = ((8750 - frequenciaMitjana - incrementFreqTC) / 8750) * 1.4;
          if (coefFreq > 0) {
            semblTC /= 1 + coefFreq;
            semblTG *= 1 + coefFreq;
          }
          else {
            coefFreq = ((frequenciaMitjana + incrementFreqTC - 9500) / 9500) * 1.55;
            if (coefFreq > 0) {
              semblTC *= 1 + coefFreq;
              semblTG /= 1 + coefFreq;
            }
          }
          coefFreq = ((8750 - frequenciaMitjana - incrementFreqTG) / 8750) * 1.4;
          if (coefFreq > 0) {
            semblTC /= 1 + coefFreq;
            semblTG *= 1 + coefFreq;
          }
          else {
            coefFreq = ((frequenciaMitjana + incrementFreqTG - 9500) / 9500) * 1.55;
            if (coefFreq > 0) {
              semblTC *= 1 + coefFreq;
              semblTG /= 1 + coefFreq;
            }
          }
        }

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = (nombreGrupsDePolsosPerSegon - 65)* 0.08 * nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblTC /= 1 + coefNGP;
            semblTG *= 1 + coefNGP;
          }
          else {
            semblTC *= 1 - coefNGP;
            semblTG /= 1 - coefNGP;
          }
        }

        semblTC /= 1 + Math.max(0, (definicioGrupsDePolsos - 0.25) * 0.5);
        semblTG *= 1 + Math.max(0, (definicioGrupsDePolsos - 0.25) * 0.5);

        if (semblTC > 1) {
          double varAux = semblTC;
          semblTC /= varAux;
          semblTG /= varAux;
        }
        if (semblTG > 1) {
          double varAux = semblTG;
          semblTC /= varAux;
          semblTG /= varAux;
        }

        semblTC = Math.max(0, Math.min(1, semblTC));
        semblTG = Math.max(0, Math.min(1, semblTG));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTH > 0.25 || semblTQ > 0.25 || semblTT > 0.25) {
          if (semblTC > semblOriginalTC) {
            semblTG *= semblOriginalTC;
            semblTG /= semblTC;
            semblTC = semblOriginalTC;
          }
          else if (semblTG > semblOriginalTG) {
            semblTC *= semblOriginalTG;
            semblTC /= semblTG;
            semblTG = semblOriginalTG;
          }
        }

        resultatSemblancaTibicinaCorsicaFairmairei = new Pair(semblTC, resultatSemblancaTibicinaCorsicaFairmairei.getSecond());
        resultatSemblancaTibicinaGarricola = new Pair(semblTG, resultatSemblancaTibicinaGarricola.getSecond());
      }

      //Tibicina garricola / Tibicina quadrisignata
      if (semblTG > 0.25 && semblTQ > 0.25) {
        double semblOriginalTG = semblTG;
        double semblOriginalTQ = semblTQ;

        semblTG /= 1 + Math.max(0, (nivellConfiancaSonCantsCurts - 0.5) * 0.6);
        semblTQ *= 1 + Math.max(0, (nivellConfiancaSonCantsCurts - 0.5) * 0.6);

        semblTG /= 1 + Math.max(0, (pendentPrimeraMeitatCantsLlargs - 1.3) * 0.3);
        semblTQ *= 1 + Math.max(0, (pendentPrimeraMeitatCantsLlargs - 1.3) * 0.3);

        semblTG /= 1 + Math.max(0, (pendentSegonaMeitatCantsLlargs - 1.2) * 0.5);
        semblTQ *= 1 + Math.max(0, (pendentSegonaMeitatCantsLlargs - 1.2) * 0.5);

        if (nombreGrupsDePolsosPerSegon > 0) {
          double coefNGP = (nombreGrupsDePolsosPerSegon - 72)* 0.09 * nivellConfiancaNombreGrupsDePolsosPerSegon;
          if (coefNGP > 0) {
            semblTG /= 1 + coefNGP;
            semblTQ *= 1 + coefNGP;
          }
          else {
            semblTG *= 1 - coefNGP;
            semblTQ /= 1 - coefNGP;
          }
        }

        if (semblTG > 1) {
          double varAux = semblTG;
          semblTG /= varAux;
          semblTQ /= varAux;
        }
        if (semblTQ > 1) {
          double varAux = semblTQ;
          semblTG /= varAux;
          semblTQ /= varAux;
        }

        semblTG = Math.max(0, Math.min(1, semblTG));
        semblTQ = Math.max(0, Math.min(1, semblTQ));

        if (semblCO > 0.25 || semblCB > 0.25 || semblCA > 0.25 || semblCBr > 0.25 || semblCC > 0.25 || semblHV > 0.25 || semblLP > 0.25 || semblTA > 0.25 || semblTP > 0.25 || semblTC > 0.25 || semblTH > 0.25 || semblTT > 0.25) {
          if (semblTG > semblOriginalTG) {
            semblTQ *= semblOriginalTG;
            semblTQ /= semblTG;
            semblTG = semblOriginalTG;
          }
          else if (semblTQ > semblOriginalTQ) {
            semblTG *= semblOriginalTQ;
            semblTG /= semblTQ;
            semblTQ = semblOriginalTQ;
          }
        }

        resultatSemblancaTibicinaGarricola = new Pair(semblTG, resultatSemblancaTibicinaGarricola.getSecond());
        resultatSemblancaTibicinaQuadrisignata = new Pair(semblTQ, resultatSemblancaTibicinaQuadrisignata.getSecond());
      }


      ArrayList<Pair<String, Double> > llistaResultats = new ArrayList<>();

      if (resultatSemblancaCicadaOrni.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaCicadaOrni.getSecond() != -1 ? "Cicada orni" : "Cicada orni*", resultatSemblancaCicadaOrni.getFirst()));
      if (resultatSemblancaCicadaBarbaraLusitanica.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaCicadaBarbaraLusitanica.getSecond() != -1 ? "Cicada barbara lusitanica" : "Cicada barbara lusitanica*", resultatSemblancaCicadaBarbaraLusitanica.getFirst()));
      if (resultatSemblancaCicadatraAtra.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaCicadatraAtra.getSecond() != -1 ? "Cicadatra atra" : "Cicadatra atra*", resultatSemblancaCicadatraAtra.getFirst()));
      if (resultatSemblancaCicadettaBrevipennis.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaCicadettaBrevipennis.getSecond() != -1 ? "Cicadetta brevipennis" : "Cicadetta brevipennis*", resultatSemblancaCicadettaBrevipennis.getFirst()));
      if (resultatSemblancaCicadettaCerdaniensis.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaCicadettaCerdaniensis.getSecond() != -1 ? "Cicadetta cerdaniensis" : "Cicadetta cerdaniensis*", resultatSemblancaCicadettaCerdaniensis.getFirst()));
      if (resultatSemblancaHilaphuraVaripes.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaHilaphuraVaripes.getSecond() != -1 ? "Hilaphura varipes" : "Hilaphura varipes*", resultatSemblancaHilaphuraVaripes.getFirst()));
```

```
    if (resultatSemblancaLyristesPlebejus.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaLyristesPlebejus.getSecond() != -1 ? "Lyristes plebejus" : "Lyristes plebejus*", resultatSemblancaLyristesPlebejus.getFirst()));
    if (resultatSemblancaTettigettalnaArgentata.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaTettigettalnaArgentata.getSecond() != -1 ? "Tettigettalna argentata" : "Tettigettalna argentata*", resultatSemblancaTettigettalnaArgentata.getFirst()));
    if (resultatSemblancaTettigettulaPygmea.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaTettigettulaPygmea.getSecond() != -1 ? "Tettigettula pygmea" : "Tettigettula pygmea*", resultatSemblancaTettigettulaPygmea.getFirst()));
    if (resultatSemblancaTibicinaCorsicaFairmairei.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaTibicinaCorsicaFairmairei.getSecond() != -1 ? "Tibicina corsica fairmairei" : "Tibicina corsica fairmairei*", resultatSemblancaTibicinaCorsicaFairmairei.getFirst()));
    if (resultatSemblancaTibicinaGarricola.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaTibicinaGarricola.getSecond() != -1 ? "Tibicina garricola" : "Tibicina garricola*", resultatSemblancaTibicinaGarricola.getFirst()));
    if (resultatSemblancaTibicinaHaematodes.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaTibicinaHaematodes.getSecond() != -1 ? "Tibicina haematodes" : "Tibicina haematodes*", resultatSemblancaTibicinaHaematodes.getFirst()));
    if (resultatSemblancaTibicinaQuadrisignata.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaTibicinaQuadrisignata.getSecond() != -1 ? "Tibicina quadrisignata" : "Tibicina quadrisignata*", resultatSemblancaTibicinaQuadrisignata.getFirst()));
    if (resultatSemblancaTibicinaTomentosa.getFirst() > 0.2) llistaResultats.add(new Pair(resultatSemblancaTibicinaTomentosa.getSecond() != -1 ? "Tibicina tomentosa" : "Tibicina tomentosa*", resultatSemblancaTibicinaTomentosa.getFirst()));

    if (!llistaResultats.isEmpty()) Collections.sort(llistaResultats, new ComparadorParelles2());


    //No aplico la restricció d'amplituds ni a la Cicadetta brevipennis ni a la Cicadetta cerdaniensis ni a la Tettigettalna argentata, atesa la tipologia característica del seu cant.
    if ((resultatSemblancaCicadaOrni.getSecond() == -2 ||
        resultatSemblancaCicadaBarbaraLusitanica.getSecond() == -2 ||
        resultatSemblancaCicadatraAtra.getSecond() == -2 ||
        resultatSemblancaCicadettaBrevipennis.getSecond() == -2 ||
        resultatSemblancaCicadettaCerdaniensis.getSecond() == -2 ||
        resultatSemblancaHilaphuraVaripes.getSecond() == -2 ||
        resultatSemblancaLyristesPlebejus.getSecond() == -2 ||
        resultatSemblancaTettigettalnaArgentata.getSecond() == -2 ||
        resultatSemblancaTettigettulaPygmea.getSecond() == -2 ||
        resultatSemblancaTibicinaCorsicaFairmairei.getSecond() == -2 ||
        resultatSemblancaTibicinaGarricola.getSecond() == -2 ||
        resultatSemblancaTibicinaHaematodes.getSecond() == -2 ||
        resultatSemblancaTibicinaQuadrisignata.getSecond() == -2 ||
        resultatSemblancaTibicinaTomentosa.getSecond() == -2) &&
        !(
          (llistaResultats.size() == 1 && llistaResultats.get(0).getFirst().compareTo("Cicadetta brevipennis") == 0 && llistaResultats.get(0).getSecond() > 0.75) ||
          (llistaResultats.size() == 1 && llistaResultats.get(0).getFirst().compareTo("Cicadetta cerdaniensis") == 0 && llistaResultats.get(0).getSecond() > 0.75) ||
          (llistaResultats.size() == 1 && llistaResultats.get(0).getFirst().compareTo("Tettigettalna argentata") == 0 && llistaResultats.get(0).getSecond() > 0.75) ||
          (
            (llistaResultats.size() == 2 && llistaResultats.get(0).getFirst().compareTo("Cicadetta cerdaniensis") == 0 && llistaResultats.get(0).getSecond() > llistaResultats.get(1).getSecond() + 0.35) ||
            (llistaResultats.size() == 2 && llistaResultats.get(0).getFirst().compareTo("Tettigettalna argentata") == 0 && llistaResultats.get(0).getSecond() > llistaResultats.get(1).getSecond() + 0.35)
          )
        )) resultatsIdentificacio += "El cant enregistrat és massa tènue per poder ser analitzat. S'hauria de gravar l'individu cantor de més a prop.";
    else if (llistaResultats.size() == 1 && llistaResultats.get(0).getFirst().compareTo("Cicada barbara lusitanica") == 0 && llistaResultats.get(0).getSecond() >= 0.2 && decibelsIncrementats > 40) {
        resultatsIdentificacio += "Cantant de fons: Cicada orni.";
    }
    else {
        if (!llistaResultats.isEmpty()) {
            resultatsIdentificacio += llistaResultats.get(0).getFirst() + ": " + String.valueOf(Math.round(llistaResultats.get(0).getSecond() * 10000) * 1.0 / 100 + "%");
            for(int i = 1; i < llistaResultats.size(); i++) {
                resultatsIdentificacio += ", " + llistaResultats.get(i).getFirst() + ": " + String.valueOf(Math.round(llistaResultats.get(i).getSecond() * 10000) * 1.0 / 100 + "%");
            }
        }
        else resultatsIdentificacio += "No s'ha trobat coincidències";
    }

    resultatsIdentificacio += "\n";

    return resultatsIdentificacio;
  }

  /**
    @throws Exception
    @throws OutOfMemoryError
    @pre ---
    @post Retorna els resultats d'identificació corresponents al fitxer entrat o als fitxers continguts per la carpeta entrada.
    @return Els resultats d'identificació corresponents al fitxer entrat o als fitxers continguts per la carpeta entrada
    @param nomFitxerOCarpeta String que conté el nom del fitxer o del directori contenidor de fitxers que s'ha d'analitzar
  */
  public static String identificaCant(String nomFitxerOCarpeta) throws OutOfMemoryError, Exception {

    //En cas que sigui un directori, calcula els resultats d'identificació per a cada fitxer .wav que aquest contingui.
    ArrayList<File> llistaFitxersSo = new ArrayList<>();
    if (nomFitxerOCarpeta.endsWith(".wav")) {
      llistaFitxersSo.add(new File(nomFitxerOCarpeta));
    }
    else {
      File dir = new File(nomFitxerOCarpeta);

      File[] matches = dir.listFiles(new FilenameFilter()
      {
       @Override
       public boolean accept(File dir, String name)
       {
         return name.endsWith(".wav");
       }
      });

      llistaFitxersSo.addAll(Arrays.asList(matches));
    }

    String resultatsIdentificacio = "";

    for(File fitxer : llistaFitxersSo) {
      resultatsIdentificacio = analitzaFitxer(fitxer, resultatsIdentificacio);
    }

    return resultatsIdentificacio;
  }

  /**
    @pre ---
    @post Retorna quatre valors: fiabilitat del càlcul de NGP (0 = bona, -1 = dolenta, -3 = no fiable), quantitat de grups de polsos + enllaços entre grups de polsos trobats, definició dels grups de polsos i, finalment, el nombre de grups de polsos per segon.
    @return Quatre valors: fiabilitat del càlcul de NGP (0 = bona, -1 = dolenta, -3 = no fiable), quantitat de grups de polsos + enllaços entre grups de polsos trobats, definició dels grups de polsos i, finalment, el nombre de grups de polsos per segon
    @param amplitudsModificades Llista d'amplituds filtrades que servirà per detectar amb precisió els grups de polsos
    @param amplitudsModificadesAbsolutesReduides Llista d'amplituds filtrades, en valor absolut i escalades que servirà per detectar els grups de polsos
    @param fs Freqüència de mostreig utilitzada
    @param llistaCants Llista de cants que hem d'analitzar
    @param indexCantMesLlarg Índex que ens indica quin dels cants de la llista llistaCants hem d'analitzar
  */
  private static Pair<Pair<Integer, Integer>, Pair<Double, Double> > trobaNGP(ArrayList<Double> amplitudsModificades, ArrayList<Double> amplitudsModificadesAbsolutesReduides, int fs, ArrayList<Pair<Double, Double> > llistaCants, int indexCantMesLlarg) {

    int analisiFiable = 0;

    //0,25 milisegons
    int pas = (int) Math.round(2*fs*0.00025);

    int seleccioInici, seleccioFinal, anteriorInici, anteriorFinal, silenciInici, silenciFinal;
    ArrayList<Pair<Double, Double> > grupsPolsos = new ArrayList<>();
    ArrayList<Pair<Double, Double> > enllacos = new ArrayList<>();

    //Cerca els grups de polsos des de l'inici del cant més llarg.
    //Afegeix 10 ms per corregir una possible desviació respecte a l'inici del cant.
    anteriorInici = (int) Math.round((llistaCants.get(indexCantMesLlarg).getFirst()+10)*4.0);
    anteriorFinal = anteriorInici + 4;
    seleccioInici = anteriorFinal + 1;
    seleccioFinal = seleccioInici + 4;

    double max1, max2, mitj1, mitj2, min1, min2;

    max1 = trobaMaximLocal(amplitudsModificades, anteriorInici*pas, anteriorFinal*pas-1);
    max2 = trobaMaximLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
```

```
    min1 = trobaMinimLocal(amplitudsModificades, anteriorInici*pas, anteriorFinal*pas-1);
    min2 = trobaMinimLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
    mitj1 = trobaMitjanaLocalAmbValorsAbsoluts(anteriorInici, anteriorFinal, amplitudsModificadesAbsolutesReduides);
    mitj2 = trobaMitjanaLocalAmbValorsAbsoluts(seleccioInici, seleccioFinal, amplitudsModificadesAbsolutesReduides);

    double coef = 2.75;
    double coef1 = 4.5;
    double coef2 = 0.5; //coef1 + coef2 = 5

    while (seleccioFinal < (int) Math.round((llistaCants.get(indexCantMesLlarg).getSecond())*4.0) - 2 && (coef1*(mitj1/mitj2)+coef2*((max1-min1)/(max2-min2))) / 5.0 < coef) {
      seleccioInici++;
      seleccioFinal++;
      anteriorInici++;
      anteriorFinal++;

      max1 = trobaMaximLocal(amplitudsModificades, anteriorInici*pas, anteriorFinal*pas-1);
      max2 = trobaMaximLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
      min1 = trobaMinimLocal(amplitudsModificades, anteriorInici*pas, anteriorFinal*pas-1);
      min2 = trobaMinimLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
      mitj1 = actualitzaMitjanaLocalAmbValorsAbsolutsPas1(anteriorInici, anteriorFinal, amplitudsModificadesAbsolutesReduides, mitj1, false);
      mitj2 = actualitzaMitjanaLocalAmbValorsAbsolutsPas1(seleccioInici, seleccioFinal, amplitudsModificadesAbsolutesReduides, mitj2, false);
    }

    silenciInici = seleccioInici;
    silenciFinal = silenciInici + 4;
    seleccioInici = silenciFinal + 1;
    seleccioFinal = seleccioInici + 4;

    //Determinat quan comença el primer silenci
    //Selecció -> primers 1.25ms del grup de polsos
    while (seleccioFinal * pas < amplitudsModificades.size() && seleccioFinal < (int) Math.round((llistaCants.get(indexCantMesLlarg).getSecond())*4.0) - 2) {
      max1 = trobaMaximLocal(amplitudsModificades, silenciInici*pas, silenciFinal*pas-1);
      max2 = trobaMaximLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
      min1 = trobaMinimLocal(amplitudsModificades, silenciInici*pas, silenciFinal*pas-1);
      min2 = trobaMinimLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
      mitj1 = trobaMitjanaLocalAmbValorsAbsoluts(silenciInici, silenciFinal, amplitudsModificadesAbsolutesReduides);
      mitj2 = trobaMitjanaLocalAmbValorsAbsoluts(seleccioInici, seleccioFinal, amplitudsModificadesAbsolutesReduides);

      while (seleccioFinal < (int) Math.round((llistaCants.get(indexCantMesLlarg).getSecond())*4.0) - 2 && (coef1*(mitj2/mitj1)+coef2*((max2-min2)/(max1-min1))) / 5.0 < coef) {
        seleccioInici++;
        seleccioFinal++;
        max2 = trobaMaximLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
        min2 = trobaMinimLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
        mitj2 = actualitzaMitjanaLocalAmbValorsAbsolutsPas1(seleccioInici, seleccioFinal, amplitudsModificadesAbsolutesReduides, mitj2, false);
      }

      if ((coef1*(mitj2/mitj1)+coef2*((max2-min2)/(max1-min1))) / 5.0 >= coef) {
        silenciFinal = seleccioInici - 1;
        anteriorInici = silenciFinal + 1;
        anteriorFinal = anteriorInici + 4;
        seleccioInici = anteriorFinal + 1;
        seleccioFinal = seleccioInici + 4;

        //Determinat quan acaba el silenci
        //Anterior -> primer 1.25ms després del grup de polsos, Selecció -> segon 1.25ms després del grup de polsos
        enllacos.add(new Pair(silenciInici*1.0, silenciFinal*1.0));

        if (seleccioFinal < (int) Math.round((llistaCants.get(indexCantMesLlarg).getSecond())*4.0)) {

          max1 = trobaMaximLocal(amplitudsModificades, anteriorInici*pas, anteriorFinal*pas-1);
          max2 = trobaMaximLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
          min1 = trobaMinimLocal(amplitudsModificades, anteriorInici*pas, anteriorFinal*pas-1);
          min2 = trobaMinimLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
          mitj1 = trobaMitjanaLocalAmbValorsAbsoluts(anteriorInici, anteriorFinal, amplitudsModificadesAbsolutesReduides);
          mitj2 = trobaMitjanaLocalAmbValorsAbsoluts(seleccioInici, seleccioFinal, amplitudsModificadesAbsolutesReduides);

          while (seleccioFinal < (int) Math.round((llistaCants.get(indexCantMesLlarg).getSecond())*4.0) - 2 && (coef1*(mitj1/mitj2)+coef2*((max1-min1)/(max2-min2))) / 5.0 < coef) {
            seleccioInici++;
            seleccioFinal++;
            anteriorFinal++;

            max1 = Math.max(max1, trobaMaximLocal(amplitudsModificades, (anteriorFinal-1)*pas, anteriorFinal*pas-1));
            max2 = trobaMaximLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
            min1 = Math.min(min1, trobaMinimLocal(amplitudsModificades, (anteriorFinal-1)*pas, anteriorFinal*pas-1));
            min2 = trobaMinimLocal(amplitudsModificades, seleccioInici*pas, seleccioFinal*pas-1);
            mitj1 = actualitzaMitjanaLocalAmbValorsAbsolutsPas1(anteriorInici, anteriorFinal, amplitudsModificadesAbsolutesReduides, mitj1, true);
            mitj2 = actualitzaMitjanaLocalAmbValorsAbsolutsPas1(seleccioInici, seleccioFinal, amplitudsModificadesAbsolutesReduides, mitj2, false);
          }

          if ((coef1*(mitj1/mitj2)+coef2*((max1-min1)/(max2-min2))) / 5.0 >= coef) {
            silenciInici = seleccioInici;
            silenciFinal = silenciInici + 4;
            seleccioInici = silenciFinal + 1;
            seleccioFinal = seleccioInici + 4;

            //Determinat quan acaba el grup de polsos
            //Silenci -> primer 1.25ms del silenci, Selecció -> Primer 1.25ms després del primer ms del silenci
            grupsPolsos.add(new Pair(anteriorInici*1.0, anteriorFinal*1.0));
          }
        }
      }
    }

    //Treu el primer silenci perquè no és fiable.
    if (enllacos.size() > 0)
      enllacos.remove(0);


    double minimaDuradaPols1 = Double.MAX_VALUE;
    double maximaDuradaPols1 = 0;
    double minimaDuradaEnllac1 = Double.MAX_VALUE;
    double maximaDuradaEnllac1 = 0;

    ArrayList<Integer> classificacioAmplitudsPolsos1 = new ArrayList<>();
    ArrayList<Integer> classificacioAmplitudsEnllacos1 = new ArrayList<>();
    ArrayList<Double> particions1 = new ArrayList<>();
    int nParticions1 = 40;
    double duradaAux1;

    double percentatgePolsosRangRivalitatTomentosa = 0;
    double percentatgeEnllacosRangRivalitatTomentosa = 0;
    double mitjanaPolsosRangRivalitatTomentosa = 0;
    double mitjanaEnllacosRangRivalitatTomentosa = 0;

    //Elimina els polsos que siguin massa llargs o massa curts (en alguns casos pot haver-ne ajuntat dos o separat un en dos de diferents).
    if (grupsPolsos.size() > 500) {

      ArrayList<Pair<Double, Double> > polsos1 = new ArrayList<>();
      polsos1.addAll(grupsPolsos);

      for(int i = 0; i < polsos1.size(); i++) {
        if (polsos1.get(i).getSecond() - polsos1.get(i).getFirst() < 4 || polsos1.get(i).getSecond() - polsos1.get(i).getFirst() > 130) {
          polsos1.remove(i);
          i--;
        }
      }

      for (Pair<Double, Double> polsos11 : polsos1) {
        duradaAux1 = polsos11.getSecond() - polsos11.getFirst();
        if (duradaAux1 > maximaDuradaPols1) maximaDuradaPols1 = duradaAux1;
        if (duradaAux1 < minimaDuradaPols1) minimaDuradaPols1 = duradaAux1;
      }

      for(int i = 1; i <= nParticions1; i++) {
        particions1.add(minimaDuradaPols1 + (maximaDuradaPols1 - minimaDuradaPols1)*i/nParticions1);
        classificacioAmplitudsPolsos1.add(0);
      }
```

```
        int iterador1;
        for (Pair<Double, Double> polsos11 : polsos1) {
          iterador1 = 0;
          duradaAux1 = polsos11.getSecond() - polsos11.getFirst();
          while (iterador1 < nParticions1-1 && duradaAux1 > particions1.get(iterador1)) {
            iterador1++;
          }
          classificacioAmplitudsPolsos1.set(iterador1, classificacioAmplitudsPolsos1.get(iterador1)+1);
        }

        int nPolsosRang = 0;
        for(int i = 0; i < classificacioAmplitudsPolsos1.size(); i++) {
          if (particions1.get(i) > 50 && particions1.get(i) < 85) {
            percentatgePolsosRangRivalitatTomentosa += classificacioAmplitudsPolsos1.get(i);
            nPolsosRang += classificacioAmplitudsPolsos1.get(i);
            mitjanaPolsosRangRivalitatTomentosa += classificacioAmplitudsPolsos1.get(i) * particions1.get(i);
          }
        }
        percentatgePolsosRangRivalitatTomentosa /= polsos1.size();
        percentatgePolsosRangRivalitatTomentosa *= 100;
        mitjanaPolsosRangRivalitatTomentosa /= nPolsosRang;
      }

      //Elimina els enllaços que siguin massa llargs o massa curts (en alguns casos pot haver-ne ajuntat dos o separat un en dos de diferents).
      if (enllacos.size() > 500) {

        ArrayList<Pair<Double, Double> > enllacos1 = new ArrayList<>();
        enllacos1.addAll(enllacos);

        for(int i = 0; i < enllacos1.size(); i++) {
          if (enllacos1.get(i).getSecond() - enllacos1.get(i).getFirst() < 4 || enllacos1.get(i).getSecond() - enllacos1.get(i).getFirst() > 130) {
            enllacos1.remove(i);
            i--;
          }
        }

        for (Pair<Double, Double> enllacos11 : enllacos1) {
          duradaAux1 = enllacos11.getSecond() - enllacos11.getFirst();
          if (duradaAux1 > maximaDuradaEnllac1) maximaDuradaEnllac1 = duradaAux1;
          if (duradaAux1 < minimaDuradaEnllac1) minimaDuradaEnllac1 = duradaAux1;
        }

        for(int i = 1; i <= nParticions1; i++) {
          particions1.add(minimaDuradaEnllac1 + (maximaDuradaEnllac1 - minimaDuradaEnllac1)*i/nParticions1);
          classificacioAmplitudsEnllacos1.add(0);
        }

        int iterador1;
        for (Pair<Double, Double> enllacos11 : enllacos1) {
          iterador1 = 0;
          duradaAux1 = enllacos11.getSecond() - enllacos11.getFirst();
          while (iterador1 < nParticions1-1 && duradaAux1 > particions1.get(iterador1)) {
            iterador1++;
          }
          classificacioAmplitudsEnllacos1.set(iterador1, classificacioAmplitudsEnllacos1.get(iterador1)+1);
        }

        int nEnllacosRang = 0;
        for(int i = 0; i < classificacioAmplitudsEnllacos1.size(); i++) {
          if (particions1.get(i) > 20 && particions1.get(i) < 50) {
            percentatgeEnllacosRangRivalitatTomentosa += classificacioAmplitudsEnllacos1.get(i);
            nEnllacosRang += classificacioAmplitudsEnllacos1.get(i);
            mitjanaEnllacosRangRivalitatTomentosa += classificacioAmplitudsEnllacos1.get(i) * particions1.get(i);
          }
        }
        percentatgeEnllacosRangRivalitatTomentosa /= enllacos1.size();
        percentatgeEnllacosRangRivalitatTomentosa *= 100;
        mitjanaEnllacosRangRivalitatTomentosa /= nEnllacosRang;
      }

      double npgRivalitatTomentosa = 0;
      //Podria tractar-se d'un cant de rivalitat de Tibicina tomentosa.
      if (percentatgePolsosRangRivalitatTomentosa > 15 && percentatgeEnllacosRangRivalitatTomentosa > 15 && mitjanaPolsosRangRivalitatTomentosa + mitjanaEnllacosRangRivalitatTomentosa > 90 && mitjanaPolsosRangRivalitatTomentosa + mitjanaEnllacosRangRivalitatTomentosa < 140) {
        analisiFiable = -2; //Aquest valor indica que es tracta d'un cant de rivalitat de Tibicina tomentosa.
        npgRivalitatTomentosa = 4000.0/(mitjanaPolsosRangRivalitatTomentosa + mitjanaEnllacosRangRivalitatTomentosa);
      }

      //Elimina enllaços i grups de polsos contigus que no poden ser correctes per les diferències excessives entre les seves durades.
      double a, b, a2, b2, c;
      for(int i = 0; i < grupsPolsos.size() - 2;i++) {
        a = grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst();
        a2 = enllacos.get(i).getSecond() - enllacos.get(i).getFirst();
        b = grupsPolsos.get(i+1).getSecond() - grupsPolsos.get(i+1).getFirst();
        b2 = enllacos.get(i+1).getSecond() - enllacos.get(i+1).getFirst();
        c = grupsPolsos.get(i+2).getSecond() - grupsPolsos.get(i+2).getFirst();
        if (a < 11 &&
          b < 11 &&
          a + a2 < 60 &&
          b + b2 < 60 &&
          a + a2 + b + b2 < 100 &&
          a + a2 + b < 60 &&
          a + b > a2 &&
          a + a2 < 20 &&
          a2 + b < 20) {
            grupsPolsos.set(i, new Pair(grupsPolsos.get(i).getFirst(), grupsPolsos.get(i+1).getSecond()));
            grupsPolsos.remove(i+1);
            enllacos.remove(i);
            i--;
        }
        else if (a2 < 3 &&
          b2 < 3 &&
          b + a2 < 56 &&
          c + b2 < 56 &&
          a2 + b + b2 + c < 100 &&
          a2 + b + b2 < 56 &&
          a2 + b2 > a + b &&
          a2 + b2 > b + c &&
          a2 + b < 20 &&
          b + b2 < 20) {
            enllacos.set(i, new Pair(enllacos.get(i).getFirst(), enllacos.get(i+1).getSecond()));
            enllacos.remove(i+1);
            grupsPolsos.remove(i+1);
            i--;
        }
      }


      ArrayList<Boolean> enllacosValids = new ArrayList<>();
      ArrayList<Boolean> polsosValids = new ArrayList<>();

      for (Pair<Double, Double> enllaco : enllacos) {
        enllacosValids.add(true);
      }

      for (Pair<Double, Double> grupsPolso : grupsPolsos) {
        polsosValids.add(true);
      }

      //Elimina enllaços massa llargs o curts (per a cigales del gènere Tibicina) -> 1ms - 16ms.
      for(int i = 0; i < enllacos.size(); i++) {
        if (enllacos.get(i).getSecond() - enllacos.get(i).getFirst() < 4 || enllacos.get(i).getSecond() - enllacos.get(i).getFirst() > 64) {
          enllacosValids.set(i, false);
        }
      }

      //Elimina polsos massa llargs o curts (per a cigales del gènere Tibicina) -> 1ms - 18ms.
      for(int i = 0; i < grupsPolsos.size(); i++) {
        if (grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst() < 4 || grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst() > 72) {
          polsosValids.set(i, false);
        }
      }

      //Marca com a no vàlids els grups de polsos i enllaços que, sumats, tinguin una diferència massa gran respecte a les parelles contígües.
      for(int i = 1; i < enllacos.size() - 1; i++) {
        if (polsosValids.get(i) &&
          ((enllacosValids.get(i) &&
            (enllacos.get(i).getSecond() - enllacos.get(i).getFirst() + grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst() < 18 ||
              enllacos.get(i).getSecond() - enllacos.get(i).getFirst() + grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst() > 112))
            || (enllacosValids.get(i+1) &&
              (enllacos.get(i+1).getSecond() - enllacos.get(i+1).getFirst() + grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst() < 18 ||
                enllacos.get(i+1).getSecond() - enllacos.get(i+1).getFirst() + grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst() > 112)))) {
```

```
                polsosValids.set(i, false);
            }
            if (enllacosValids.get(i) &&
                ((polsosValids.get(i-1) &&
                    (grupsPolsos.get(i-1).getSecond() - grupsPolsos.get(i-1).getFirst() + enllacos.get(i).getSecond() - enllacos.get(i).getFirst() < 18 ||
                    grupsPolsos.get(i-1).getSecond() - grupsPolsos.get(i-1).getFirst() + enllacos.get(i).getSecond() - enllacos.get(i).getFirst() > 112))
                || (polsosValids.get(i) &&
                    (grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst() + enllacos.get(i).getSecond() - enllacos.get(i).getFirst() < 18 ||
                    grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst() + enllacos.get(i).getSecond() - enllacos.get(i).getFirst() > 112)))) {

                enllacosValids.set(i, false);
            }
        }

        //Elimina els enllaços i els grups de polsos no vàlids. No es fa directament per evitar que interfereixi a l'hora de comparar durades de grups de polsos i enllaços contígües.
        for(int i = 0; i < enllacos.size(); i++) {
            if (!enllacosValids.get(i)) {
                enllacos.remove(i);
                enllacosValids.remove(i);
                i--;
            }
        }

        for(int i = 0; i < grupsPolsos.size(); i++) {
            if (!polsosValids.get(i)) {
                grupsPolsos.remove(i);
                polsosValids.remove(i);
                i--;
            }
        }

        //Perquè l'anàlisi sigui fiable, hi ha d'haver un mínim de grups de polsos i enllaços entre aquests trobats i correctes.
        if (grupsPolsos.size() < 40 || enllacos.size() < 40) analisiFiable = -1;

        int iterador;
        int nParticions;
        double minimaDuradaPols, maximaDuradaPols, minimaDuradaEnllac, maximaDuradaEnllac, duradaAux;

        minimaDuradaPols = Double.MAX_VALUE;
        maximaDuradaPols = 0;

        ArrayList<Integer> classificacioAmplitudsPolsos = new ArrayList<>();
        ArrayList<Double> particions = new ArrayList<>();
        nParticions = 40;

        //Classifica els grups de polsos per eliminar els incorrectes (els que estiguin massa allunyats dels valors de la majoria).
        if (grupsPolsos.size() > 5) {

            for (Pair<Double, Double> grupsPolso : grupsPolsos) {
                duradaAux = grupsPolso.getSecond() - grupsPolso.getFirst();
                if (duradaAux > maximaDuradaPols) maximaDuradaPols = duradaAux;
                if (duradaAux < minimaDuradaPols) minimaDuradaPols = duradaAux;
            }

            for(int i = 1; i <= nParticions; i++) {
                particions.add(minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*i/nParticions);
                classificacioAmplitudsPolsos.add(0);
            }

            for (Pair<Double, Double> grupsPolso : grupsPolsos) {
                iterador = 0;
                duradaAux = grupsPolso.getSecond() - grupsPolso.getFirst();
                while (iterador < nParticions-1 && duradaAux > particions.get(iterador)) {
                    iterador++;
                }
                classificacioAmplitudsPolsos.set(iterador, classificacioAmplitudsPolsos.get(iterador)+1);
            }

            //Elimina els pics incorrectes. Quan té dos pics separats, és evident que s'ha de basar en un dels dos, no en la mitjana, per això elimina el menys fiable.
            ArrayList<Integer> llistaPics;
            ArrayList<Double> mitjanesPics;
            double mitjana1, mitjana2;
            int nValorsMitjana1, nValorsMitjana2;
            for(int k = 0; k < 2; k++) {
                llistaPics = new ArrayList<>();
                for(int l = 0; l < classificacioAmplitudsPolsos.size(); l++) {
                    mitjana1 = 0;
                    mitjana2 = 0;
                    nValorsMitjana1 = 0;
                    nValorsMitjana2 = 0;
                    for(int m = Math.max(l-4, 0); m < l; m++) {
                        mitjana1 += classificacioAmplitudsPolsos.get(m);
                        nValorsMitjana1++;
                    }
                    if (nValorsMitjana1 > 0) mitjana1 /= nValorsMitjana1;
                    for(int m = Math.min(l+1, classificacioAmplitudsPolsos.size()-1); m < Math.min(l+4, classificacioAmplitudsPolsos.size()-1); m++) {
                        mitjana2 += classificacioAmplitudsPolsos.get(m);
                        nValorsMitjana2++;
                    }
                    if (nValorsMitjana2 > 0) mitjana2 /= nValorsMitjana2;
                    if (classificacioAmplitudsPolsos.get(l) > mitjana1 && classificacioAmplitudsPolsos.get(l) > mitjana2) llistaPics.add(l);
                }

                mitjanesPics = new ArrayList<>();

                for (Integer llistaPic : llistaPics) {
                    int inici1, fi1;
                    mitjana1 = 0;
                    inici1 = Math.max(llistaPic - 4, 0);
                    fi1 = Math.min(llistaPic + 4, classificacioAmplitudsPolsos.size()-1);
                    for(int n = inici1; n <= fi1; n++) {
                        mitjana1 += classificacioAmplitudsPolsos.get(n);
                    }
                    mitjana1 /= (fi1 - inici1 + 1);
                    mitjanesPics.add(mitjana1);
                }

                for(int l = 0; l < llistaPics.size(); l++) {
                    for(int m = l + 1; m < llistaPics.size(); m++) {
                        if (llistaPics.get(m) < llistaPics.get(l)+5) {
                            if (mitjanesPics.get(l) > mitjanesPics.get(m)) {
                                llistaPics.remove(m);
                                mitjanesPics.remove(m);
                                m--;
                            }
                            else {
                                llistaPics.remove(l);
                                mitjanesPics.remove(l);
                                l--;
                                m = llistaPics.size(); //Acaba el bucle de comparacions del pic l amb altres pics.
                            }
                        }
                    }
                }

                //Elimina els pics incompatibles amb altres pics més importants.
                for(int l = 0; l < llistaPics.size(); l++) {
                    for(int m = l + 1; m < llistaPics.size(); m++) {
                        if ((minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*llistaPics.get(m)/nParticions) - (minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*llistaPics.get(l)/nParticions) > 15) {
                            int inici1, fi1, inici2, fi2;
                            inici1 = Math.max(llistaPics.get(l)-4, 0);
                            inici2 = Math.max(llistaPics.get(m)-4, 0);
                            fi1 = Math.min(llistaPics.get(l)+4, classificacioAmplitudsPolsos.size()-1);
                            fi2 = Math.min(llistaPics.get(m)+4, classificacioAmplitudsPolsos.size()-1);


                            if (mitjanesPics.get(l) > mitjanesPics.get(m)) {
                                for(int n = inici2; n <= fi2; n++) {
                                    classificacioAmplitudsPolsos.set(n, 0);
                                }
                                if (mitjanesPics.get(m) > grupsPolsos.size()*0.0325 && mitjanesPics.get(l) < mitjanesPics.get(m)*1.3) analisiFiable = -1;
                                mitjanesPics.set(m, 0.0);
                            }
                            else {
                                for(int n = inici1; n <= fi1; n++) {
                                    classificacioAmplitudsPolsos.set(n, 0);
                                }
                                if (mitjanesPics.get(l) > grupsPolsos.size()*0.0325 && mitjanesPics.get(m) < mitjanesPics.get(l)*1.3) analisiFiable = -1;
                                mitjanesPics.set(l, 0.0);
                                m = llistaPics.size(); //Acaba el bucle de comparacions del pic l amb altres pics.
                            }
                        }
                    }
                }
            }


            //Troba la mitjana de NGP als trams d'amplitud on n'hi ha més.
            int maxim1, maxim2, maxim3, maxim4, maxim5;
            maxim1 = -1;
            maxim2 = -1;
```

```
        maxim3 = -1;
        maxim4 = -1;
        maxim5 = -1;


        double mitjMax;

        for(int i = 0; i < classificacioAmplitudsPolsos.size(); i++) {
          if (maxim1 < 0 || classificacioAmplitudsPolsos.get(i) > classificacioAmplitudsPolsos.get(maxim1)) {
            maxim5 = maxim4;
            maxim4 = maxim3;
            maxim3 = maxim2;
            maxim2 = maxim1;
            maxim1 = i;
          }
          else if (maxim2 < 0 || classificacioAmplitudsPolsos.get(i) > classificacioAmplitudsPolsos.get(maxim2)) {
            maxim5 = maxim4;
            maxim4 = maxim3;
            maxim3 = maxim2;
            maxim2 = i;
          }
          else if (maxim3 < 0 || classificacioAmplitudsPolsos.get(i) > classificacioAmplitudsPolsos.get(maxim3)) {
            maxim5 = maxim4;
            maxim4 = maxim3;
            maxim3 = i;
          }
          else if (maxim4 < 0 || classificacioAmplitudsPolsos.get(i) > classificacioAmplitudsPolsos.get(maxim4)) {
            maxim5 = maxim4;
            maxim4 = i;
          }
          else if (maxim5 < 0 || classificacioAmplitudsPolsos.get(i) > classificacioAmplitudsPolsos.get(maxim5)) {
            maxim5 = i;
          }
        }

        if (maxim1 == maxim2 || maxim2 == maxim3 || maxim3 == maxim4 || maxim4 == maxim5 || classificacioAmplitudsPolsos.get(maxim1) + classificacioAmplitudsPolsos.get(maxim2) + classificacioAmplitudsPolsos.get(maxim3) + classificacioAmplitudsPolsos.get(maxim4) + classificacioAmplitudsPolsos.get(maxim5) < 25) analisiFiable = -1;

        mitjMax = (classificacioAmplitudsPolsos.get(maxim1)*(minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*maxim1/nParticions) +
                classificacioAmplitudsPolsos.get(maxim2)*(minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*maxim2/nParticions) +
                classificacioAmplitudsPolsos.get(maxim3)*(minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*maxim3/nParticions) +
                classificacioAmplitudsPolsos.get(maxim4)*(minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*maxim4/nParticions) +
                classificacioAmplitudsPolsos.get(maxim5)*(minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*maxim5/nParticions)) /
                (classificacioAmplitudsPolsos.get(maxim1) + classificacioAmplitudsPolsos.get(maxim2) + classificacioAmplitudsPolsos.get(maxim3) + classificacioAmplitudsPolsos.get(maxim4) + classificacioAmplitudsPolsos.get(maxim5));


        //Defineix quins polsos considerarà vàlids a partir de la seva posició dins de la classificació de grups de polsos per trams d'amplitud.
        double coef3 = 0.2;
        double coef4 = 1.0;

        int maximPolsosEnFragment = 0;
        for (Integer classificacioAmplitudsPolso : classificacioAmplitudsPolsos) {
          if (classificacioAmplitudsPolso > maximPolsosEnFragment) {
            maximPolsosEnFragment = classificacioAmplitudsPolso;
          }
        }

        int nIniciPolsos = 0;
        int sumatori = 0;
        iterador = nIniciPolsos;
        while (iterador < classificacioAmplitudsPolsos.size() - 1 && !(classificacioAmplitudsPolsos.get(iterador) > maximPolsosEnFragment*coef3)) {
          sumatori += classificacioAmplitudsPolsos.get(iterador);
          iterador++;
        }
        nIniciPolsos = iterador;
        int nFinalPolsos = nIniciPolsos;
        sumatori += classificacioAmplitudsPolsos.get(iterador);
        iterador++;
        while (iterador < classificacioAmplitudsPolsos.size()&& !(sumatori > grupsPolsos.size()*coef4)) {
          sumatori += classificacioAmplitudsPolsos.get(iterador);
          if (classificacioAmplitudsPolsos.get(iterador) > maximPolsosEnFragment*coef3 && (minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*iterador/nParticions) < mitjMax*2.5 && (minimaDuradaPols + (maximaDuradaPols - minimaDuradaPols)*iterador/nParticions) > mitjMax/2.5) nFinalPolsos = iterador;
          iterador++;
        }


        //Elimina els que no considera vàlids.
        for(int i = 0; i < grupsPolsos.size(); i++) {
          iterador = 0;
          duradaAux = grupsPolsos.get(i).getSecond() - grupsPolsos.get(i).getFirst();
          while (iterador < nParticions-1 && duradaAux > particions.get(iterador)) {
            iterador++;
          }
          if (iterador < nIniciPolsos || iterador > nFinalPolsos) {
            grupsPolsos.remove(i);
            i--;
          }
        }

        minimaDuradaEnllac = Double.MAX_VALUE;
        maximaDuradaEnllac = 0;

        ArrayList<Integer> classificacioAmplitudsEnllacos = new ArrayList<>();
        particions = new ArrayList<>();
        nParticions = 40;

        //Classifica els enllaços per eliminar els incorrectes (els que estiguin massa allunyats dels valors de la majoria).
        if (enllacos.size() > 5) {

          for (Pair<Double, Double> enllaco : enllacos) {
            duradaAux = enllaco.getSecond() - enllaco.getFirst();
            if (duradaAux > maximaDuradaEnllac) maximaDuradaEnllac = duradaAux;
            if (duradaAux < minimaDuradaEnllac) minimaDuradaEnllac = duradaAux;
          }

          for(int i = 1; i <= nParticions; i++) {
            particions.add(minimaDuradaEnllac + (maximaDuradaEnllac - minimaDuradaEnllac)*i/nParticions);
            classificacioAmplitudsEnllacos.add(0);
          }

          for (Pair<Double, Double> enllaco : enllacos) {
            iterador = 0;
            duradaAux = enllaco.getSecond() - enllaco.getFirst();
            while (iterador < nParticions-1 && duradaAux > particions.get(iterador)) {
              iterador++;
            }
            classificacioAmplitudsEnllacos.set(iterador, classificacioAmplitudsEnllacos.get(iterador)+1);
          }


          //Elimina els pics incorrectes. Quan té dos pics separats, és evident que s'ha de basar en un dels dos, no en la mitjana, per això elimina el menys fiable.
          ArrayList<Integer> llistaPics;
          ArrayList<Double> mitjanesPics;
          double mitjana1, mitjana2;
          int nValorsMitjana1, nValorsMitjana2;
          for(int k = 0; k < 2; k++) {
            llistaPics = new ArrayList<>();
            for(int l = 0; l < classificacioAmplitudsEnllacos.size(); l++) {
              mitjana1 = 0;
              mitjana2 = 0;
              nValorsMitjana1 = 0;
              nValorsMitjana2 = 0;
              for(int m = Math.max(l-4, 0); m < l; m++) {
                mitjana1 += classificacioAmplitudsEnllacos.get(m);
                nValorsMitjana1++;
              }
              if (nValorsMitjana1 > 0) mitjana1 /= nValorsMitjana1;
              for(int m = Math.min(l+1, classificacioAmplitudsEnllacos.size()-1); m < Math.min(l+4, classificacioAmplitudsEnllacos.size()-1); m++) {
                mitjana2 += classificacioAmplitudsEnllacos.get(m);
                nValorsMitjana2++;
              }
              if (nValorsMitjana2 > 0) mitjana2 /= nValorsMitjana2;
              if (classificacioAmplitudsEnllacos.get(l) > mitjana1 && classificacioAmplitudsEnllacos.get(l) > mitjana2) llistaPics.add(l);
            }

            mitjanesPics = new ArrayList<>();

            for (Integer llistaPic : llistaPics) {
              int inici1, fi1;
              mitjana1 = 0;
              inici1 = Math.max(llistaPic - 4, 0);
              fi1 = Math.min(llistaPic + 4, classificacioAmplitudsEnllacos.size()-1);
              for(int n = inici1; n <= fi1; n++) {
```

```
            mitjana1 += classificacioAmplitudsEnllacos.get(n);
        }
        mitjana1 /= (fi1 - inici1 + 1);
        mitjanesPics.add(mitjana1);
      }

      for(int l = 0; l < llistaPics.size(); l++) {
        for(int m = l + 1; m < llistaPics.size(); m++) {
          if (llistaPics.get(m) < llistaPics.get(l)+5) {
            if (mitjanesPics.get(l) > mitjanesPics.get(m)) {
              llistaPics.remove(m);
              mitjanesPics.remove(m);
              m--;
            }
            else {
              llistaPics.remove(l);
              mitjanesPics.remove(l);
              l--;
              m = llistaPics.size(); //Acaba el bucle de comparacions del pic l amb
altres pics.
            }
          }
        }
      }

      //Elimina els pics incompatibles amb altres pics més importants.
      for(int l = 0; l < llistaPics.size(); l++) {
        for(int m = l + 1; m < llistaPics.size(); m++) {
          if ((minimaDuradaEnllac + (maximaDuradaEnllac -
minimaDuradaEnllac)*llistaPics.get(m)/nParticions) - (minimaDuradaEnllac +
(maximaDuradaEnllac - minimaDuradaEnllac)*llistaPics.get(l)/nParticions) > 15) {
            int inici1, fi1, inici2, fi2;
            inici1 = Math.max(llistaPics.get(l)-4, 0);
            inici2 = Math.max(llistaPics.get(m)-4, 0);
            fi1 = Math.min(llistaPics.get(l)+4, classificacioAmplitudsEnllacos.size()-
1);
            fi2 = Math.min(llistaPics.get(m)+4,
classificacioAmplitudsEnllacos.size()-1);


            if (mitjanesPics.get(l) > mitjanesPics.get(m)) {
              for(int n = inici2; n <= fi2; n++) {
                classificacioAmplitudsEnllacos.set(n, 0);
              }
              if (mitjanesPics.get(m) > enllacos.size()*0.0325 &&
mitjanesPics.get(l) < mitjanesPics.get(m)*1.3) analisiFiable = -1;
              mitjanesPics.set(m, 0.0);
            }
            else {
              for(int n = inici1; n <= fi1; n++) {
                classificacioAmplitudsEnllacos.set(n, 0);
              }
              if (mitjanesPics.get(l) > enllacos.size()*0.0325 &&
mitjanesPics.get(m) < mitjanesPics.get(l)*1.3) analisiFiable = -1;
              mitjanesPics.set(l, 0.0);
              m = llistaPics.size(); //Acaba el bucle de comparacions del pic l amb
altres pics.
            }
          }
        }
      }


    //Troba la mitjana d'enllaços als trams d'amplitud on n'hi ha més.
    int maxim1, maxim2, maxim3, maxim4, maxim5;
    maxim1 = -1;
    maxim2 = -1;
    maxim3 = -1;
    maxim4 = -1;
    maxim5 = -1;

    double mitjMax;

    for(int i = 0; i < classificacioAmplitudsEnllacos.size(); i++) {
      if (maxim1 < 0 || classificacioAmplitudsEnllacos.get(i) >
classificacioAmplitudsEnllacos.get(maxim1)) {
        maxim5 = maxim4;
        maxim4 = maxim3;
        maxim3 = maxim2;
        maxim2 = maxim1;
        maxim1 = i;
      }
      else if (maxim2 < 0 || classificacioAmplitudsEnllacos.get(i) >
classificacioAmplitudsEnllacos.get(maxim2)) {
        maxim5 = maxim4;
        maxim4 = maxim3;
        maxim3 = maxim2;
        maxim2 = i;
      }
      else if (maxim3 < 0 || classificacioAmplitudsEnllacos.get(i) >
classificacioAmplitudsEnllacos.get(maxim3)) {
        maxim5 = maxim4;
        maxim4 = maxim3;
        maxim3 = i;
      }
      else if (maxim4 < 0 || classificacioAmplitudsEnllacos.get(i) >
classificacioAmplitudsEnllacos.get(maxim4)) {
        maxim5 = maxim4;
        maxim4 = i;
      }
      else if (maxim5 < 0 || classificacioAmplitudsEnllacos.get(i) >
classificacioAmplitudsEnllacos.get(maxim5)) {
        maxim5 = i;
      }
    }

    if (maxim1 == maxim2 || maxim2 == maxim3 || maxim3 == maxim4 || maxim4
== maxim5 || classificacioAmplitudsEnllacos.get(maxim1) +
classificacioAmplitudsEnllacos.get(maxim2) +
classificacioAmplitudsEnllacos.get(maxim3) +
classificacioAmplitudsEnllacos.get(maxim4) +
classificacioAmplitudsEnllacos.get(maxim5) < 25) analisiFiable = -1;

    mitjMax = (classificacioAmplitudsEnllacos.get(maxim1)*(minimaDuradaEnllac
+ (maximaDuradaEnllac - minimaDuradaEnllac)*maxim1/nParticions) +
        classificacioAmplitudsEnllacos.get(maxim2)*(minimaDuradaEnllac +
(maximaDuradaEnllac - minimaDuradaEnllac)*maxim2/nParticions) +
        classificacioAmplitudsEnllacos.get(maxim3)*(minimaDuradaEnllac +
(maximaDuradaEnllac - minimaDuradaEnllac)*maxim3/nParticions) +
        classificacioAmplitudsEnllacos.get(maxim4)*(minimaDuradaEnllac +
(maximaDuradaEnllac - minimaDuradaEnllac)*maxim4/nParticions) +
        classificacioAmplitudsEnllacos.get(maxim5)*(minimaDuradaEnllac +
(maximaDuradaEnllac - minimaDuradaEnllac)*maxim5/nParticions)) /
        (classificacioAmplitudsEnllacos.get(maxim1) +
classificacioAmplitudsEnllacos.get(maxim2) +
classificacioAmplitudsEnllacos.get(maxim3) +
classificacioAmplitudsEnllacos.get(maxim4) +
classificacioAmplitudsEnllacos.get(maxim5));


    //Decideix quins enllacos considerarà vàlids a partir de la seva posició dins de la
classificació d'enllaços per trams d'amplitud.
    double coef3 = 0.2;
    double coef4 = 1.0;
    int maximEnllacosEnFragment = 0;
    for (Integer classificacioAmplitudsEnllaco : classificacioAmplitudsEnllacos) {
      if (classificacioAmplitudsEnllaco > maximEnllacosEnFragment) {
        maximEnllacosEnFragment = classificacioAmplitudsEnllaco;
      }
    }

    int nIniciEnllacos = 0;
    int sumatori = 0;
    iterador = nIniciEnllacos;
    while (iterador < classificacioAmplitudsEnllacos.size() - 1 &&
!(classificacioAmplitudsEnllacos.get(iterador) > maximEnllacosEnFragment*coef3)) {
      sumatori += classificacioAmplitudsEnllacos.get(iterador);
      iterador++;
    }
    nIniciEnllacos = iterador;
    int nFinalEnllacos = nIniciEnllacos;
    sumatori += classificacioAmplitudsEnllacos.get(iterador);
    iterador++;
    while (iterador < classificacioAmplitudsEnllacos.size()&& !(sumatori >
enllacos.size()*coef4)) {
      sumatori += classificacioAmplitudsEnllacos.get(iterador);
      if (classificacioAmplitudsEnllacos.get(iterador) >
maximEnllacosEnFragment*coef3 && (minimaDuradaEnllac + (maximaDuradaEnllac -
minimaDuradaEnllac)*iterador/nParticions) < mitjMax*2.5 && (minimaDuradaEnllac +
(maximaDuradaEnllac - minimaDuradaEnllac)*iterador/nParticions) > mitjMax/2.5)
nFinalEnllacos = iterador;
      iterador++;
    }

    //Elimina els que no considera vàlids.
    for(int i = 0; i < enllacos.size(); i++) {
      iterador = 0;
      duradaAux = enllacos.get(i).getSecond() - enllacos.get(i).getFirst();
      while (iterador < nParticions-1 && duradaAux > particions.get(iterador)) {
        iterador++;
      }
      if (iterador < nIniciEnllacos || iterador > nFinalEnllacos) {
        enllacos.remove(i);
        i--;
      }
    }
  }

  //Calcula la mitjana de NGP a partir de la durada mitjana dels grups de polsos i els
enllaços que els uneixen.
```

```
    double mitjanaDuradaGrupPolsos = 0;
    double mitjanaDuradaEnllac = 0;

    for (Pair<Double, Double> grupsPolsos1 : grupsPolsos) {
      mitjanaDuradaGrupPolsos += grupsPolsos1.getSecond() - grupsPolsos1.getFirst() + 1;
    }

    if (grupsPolsos.size() > 0)
      mitjanaDuradaGrupPolsos /= grupsPolsos.size();
    else
      mitjanaDuradaGrupPolsos = 0;

    for (Pair<Double, Double> enllacos1 : enllacos) {
      mitjanaDuradaEnllac += enllacos1.getSecond() - enllacos1.getFirst() + 1;
    }

    if (enllacos.size() > 0)
      mitjanaDuradaEnllac /= enllacos.size();
    else
      mitjanaDuradaEnllac = 0;

    double nombreGrupsDePolsosPerSegon;
    if (mitjanaDuradaGrupPolsos + mitjanaDuradaEnllac > 0)
      nombreGrupsDePolsosPerSegon = 4000/(mitjanaDuradaGrupPolsos + mitjanaDuradaEnllac);
    else
      nombreGrupsDePolsosPerSegon = 0;

    //Amb tan poques mostres, la fiabilitat del resultat és quasi nul·la.
    if (enllacos.size() < 10 || grupsPolsos.size() < 10) analisiFiable = -3;

    if (analisiFiable == -2) nombreGrupsDePolsosPerSegon = npgRivalitatTomentosa;

    //Calcula la definició a partir dels grups de polsos i enllaços vàlids que ha pogut trobar per unitat de temps.
    double definicioGrupsDePolsos;
    if (nombreGrupsDePolsosPerSegon > 0) {
      double duradaEnms = llistaCants.get(indexCantMesLlarg).getSecond() - llistaCants.get(indexCantMesLlarg).getFirst();
      double nGrupsDePolsosEsperats = duradaEnms / nombreGrupsDePolsosPerSegon;
      definicioGrupsDePolsos = Math.max(0, Math.min(1, (enllacos.size() + grupsPolsos.size()) / (nGrupsDePolsosEsperats*2000.0)));
    }
    else definicioGrupsDePolsos = 0.0;


    return new Pair(new Pair(analisiFiable, enllacos.size() + grupsPolsos.size()), new Pair(definicioGrupsDePolsos, nombreGrupsDePolsosPerSegon));
  }

  /**
    @throws OutOfMemoryError
    @throws Exception
    @pre ---
    @post Mostra els resultats d'identificació corresponents a un fitxer o als fitxers continguts per una carpeta.
    @param args String que conté el nom del fitxer o del directori contenidor de fitxers que s'ha d'analitzar
  */
  public static void main(String[] args) throws OutOfMemoryError, Exception {

    String fitxerOCarpeta = "";
    if (args.length > 0) {
      fitxerOCarpeta = args[0];
    }
    else {
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicada\\Cicada orni\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicada\\Cicada barbara lusitanica\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicadatra atra\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicadetta\\Cicadetta brevipennis\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicadetta\\Cicadetta cerdaniensis\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Hilaphura varipes\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Lyristes plebejus\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tettigettalna argentata\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tettigettula pygmea\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina corsica fairmairei\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina garricola\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina haematodes\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina quadrisignata\\";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina tomentosa\\";


      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicada orni__12.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicada orni__excitada.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicadatra atra___3__1.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Lyristes plebejus__2.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tettigettalna argentata_Reclam.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tettigettalna argentata__4.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tettigettalna argentata.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tettigettula pygmea__3.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina corsica fairmairei 2.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina quadrisignata 3.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina tomentosa__10.wav";


      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicada\\Cicada orni\\Cicada orni_Concert.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicada\\Cicada barbara lusitanica\\Cicada barbara__.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicadatra atra\\Cicadatra atra__soroll.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicadetta\\Cicadetta brevipennis\\Cicadetta brevipennis-2.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicadetta\\Cicadetta cerdaniensis\\Cicadetta cerdaniensis.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Hilaphura varipes\\Hilaphura varipes.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tettigettalna argentata\\Tettigettalna argentata__.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tettigettula pygmea\\Tettigettula pygmea___4__2.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina garricola\\Tibicina garricola.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina quadrisignata\\Tibicina quadrisignata__soroll 4.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina corsica fairmairei\\Tibicina sp.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina tomentosa\\Tibicina tomentosa__soroll.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Lyristes plebejus\\Lyristes plebejus.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Cicada\\Cicada barbara lusitanica\\Cicada barbara lusitanica__5__11.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina haematodes\\Tibicina haematodes__soroll.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Per analitzar\\Tibicina\\Tibicina sp\\Tibicina sp - 04-07-16.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Vídeos\\6.wav";
      //fitxerOCarpeta = "C:\\Users\\David\\Desktop\\Cigales\\Ortòpters\\Ortòpter__2.wav";
    }

    String resultatsIdentificacio;

    if (fitxerOCarpeta.compareTo("") != 0) resultatsIdentificacio = identificaCant(fitxerOCarpeta);
    else resultatsIdentificacio = "No s'ha especificat cap fitxer.";

    System.out.print(resultatsIdentificacio);
  }
}
```

ComparadorParelles.java:

```
package cigales;

import Parella.Pair;
import java.util.Comparator;

/**
  @brief ComparadorParelles és una classe auxiliar que permet comparar dues parelles de reals. La utilitzo per comparar durades de cants, les quals determina
  * restant el primer component de la parella (inici del cant) al segon (final del cant). Retorna -1 en cas que el primer cant sigui més llarg, 1 si el més llarg és el segon i
  * 0 si tots dos duren el mateix.
*/

public class ComparadorParelles implements Comparator<Pair<Double, Double> > {

  /**
    @pre ---
    @post Retorna -1 en cas que el primer cant sigui més llarg (segon component de la parella - primer component), 1 si el més llarg és el segon i 0 si tots dos duren el mateix.
    @return -1 en cas que el primer cant sigui més llarg, 1 si el més llarg és el segon i 0 si tots dos duren el mateix
    @param a Primera parella
    @param b Segona parella
  */
  @Override
  public int compare(Pair<Double, Double> a, Pair<Double, Double> b) {

    double duradaA = a.getSecond() - a.getFirst();
    double duradaB = b.getSecond() - b.getFirst();

    if (duradaA > duradaB) return -1;
    else if (duradaA == duradaB) return 0;
    else return 1;
  }
}
```

ComparadorParelles2.java:

```
package cigales;

import Parella.Pair;
import java.util.Comparator;

/**
  @brief ComparadorParelles2 és una classe auxiliar que permet comparar dues parelles amb components de tipus String i real. La utilitzo per comparar resultats de semblança amb
  * les espècies de cigala que tracta l'aplicació. Retorna -1 en cas que el primer resultat sigui millor que el segon, 1 si el millor és el segon i 0 si s'ha trobat el mateix
  * nivell de semblança per a totes dues espècies.
*/

public class ComparadorParelles2 implements Comparator<Pair<String, Double> > {

  /**
    @pre ---
    @post Retorna -1 en cas que el primer resultat sigui millor que el segon (comparant els segons components de la parella), 1 si el millor és el segon i 0 si s'ha trobat el mateix nivell de semblança per a totes dues espècies.
    @return -1 en cas que el primer resultat sigui millor que el segon (comparant els segons components de la parella), 1 si el millor és el segon i 0 si s'ha trobat el mateix nivell de semblança per a totes dues espècies
    @param a Primera parella
    @param b Segona parella
  */
  @Override
  public int compare(Pair<String, Double> a, Pair<String, Double> b) {

    if (a.getSecond() < b.getSecond()) return 1;
    else if (a.getSecond() == b.getSecond()) return 0;
    else return -1;
  }
}
```

DadesAudio.java:

```
package cigales;

import java.io.IOException;
import java.io.InputStream;

/**
  @brief DadesAudio és una classe que llegeix i guarda les dades d'àudio d'un fitxer .wav. S'utilitza per obtenir els valors d'amplitud en domini temporal del fitxer que s'analitza.
*/

public class DadesAudio extends InputStream {

  /** Guarda el flux d'àudio del fitxer WAV que es vol analitzar. */
  private final InputStream flux;

  /** Guarda les dades corresponents al format del fitxer WAV que es vol analitzar. */
  private final FormatWav format;

  /** Guarda la mida de quadre del fitxer WAV que es vol analitzar (quadre = "frame" en anglès). */
  private int midaDeQuadre;

  /** Guarda la posició de lectura actual dins del flux d'àudio del fitxer WAV que es vol analitzar. */
  private long posicio;

  /** Guarda la llargada en bytes del flux d'àudio del fitxer WAV que es vol analitzar. */
  private final long llargadaEnBytes;

  /**
    @pre ---
    @post Crea un nou objecte DadesAudio el flux i el format entrats per paràmetre.
    @param fluxEntrada Flux d'àudio del fitxer WAV que es vol analitzar
    @param format Format del fitxer WAV que es vol analitzar
  */
  public DadesAudio(InputStream fluxEntrada, FormatWav format) {
    this.flux = fluxEntrada;
    this.format = format;
    this.midaDeQuadre = format.obtinguesMidaDeQuadre();
    if (this.midaDeQuadre < 1) {
      this.midaDeQuadre = 1;
    }
    this.llargadaEnBytes = obtinguesLlargadaEnBytes((FormatWav)format);
    this.posicio = 0;
  }

  /**
    @pre ---
    @post Retorna la llargada, en bytes, del flux d'àudio del fitxer WAV que es vol analitzar.
    @param format Format del fitxer WAV que es vol analitzar
    @return La llargada, en bytes, del flux d'àudio del fitxer WAV que es vol analitzar
  */
  public static long obtinguesLlargadaEnBytes(FormatWav format) {
    int midaDeQuadre = format.obtinguesMidaDeQuadre();
    if (format.obtinguesLlargadaEnQuadres() >= 0 && midaDeQuadre >= 1) {
      return format.obtinguesLlargadaEnQuadres() * (long)midaDeQuadre;
    }
    return -1;
  }

  /**
    @pre ---
    @post Retorna el nombre de quadres que té el fitxer WAV llegit.
    @return El nombre de quadres que té el fitxer WAV llegit
  */
  public int obtinguesLlargadaEnQuadres() {
    return this.format.obtinguesLlargadaEnQuadres();
  }

  /**
    @pre ---
    @post Retorna el format del fitxer WAV llegit.
    @return El format del fitxer WAV llegit
  */
  public FormatWav obtinguesFormat() {
    return this.format;
  }

  /**
    @pre ---
    @post Llegeix el primer valor, des de la posició de lectura actual, del flux del fitxer WAV llegit.
    @return El primer valor, des de la posició de lectura actual, del flux del fitxer WAV llegit, o bé -1 en cas que no quedin més valors per llegir
    @throws IOException La mida de quadre és diferent de 1
  */
  @Override
  public int read() throws IOException {
    if (this.midaDeQuadre != 1) {
      throw new IOException("La mida de quadre ha de ser 1 per poder llegir un sol byte.");
    }
    if (this.sHaArribatAlFinalDelFitxer()) {
```

```
            return -1;
        }
        int nByte = this.flux.read();
        if (nByte != -1) {
            this.posicio++;
        }
        return nByte;
    }

    /**
        @pre ---
        @post Llegeix el primers dadesAB.length valors, des de la posició de lectura actual, del flux del fitxer WAV llegit, i els guarda a la llista de bytes ja esmentada.
        @param dadesAB Llista de bytes que s'ha d'omplir amb els valors llegits del flux d'àudio de l'objecte actual
        @return El nombre de bytes que s'ha pogut llegir
        @throws IOException La llargada de valors que es vol llegir no és múltiple de la mida de quadre
    */
    @Override
    public int read(byte[] dadesAB) throws IOException {
        return this.read(dadesAB, 0, dadesAB.length);
    }

    /**
        @pre ---
        @post Llegeix el primers dadesAB.length valors, des de la posició de lectura actual, del flux del fitxer WAV llegit, i els guarda a la llista de bytes ja esmentada.
        @param dadesAB Llista de bytes que s'ha d'omplir amb els valors llegits del flux d'àudio de l'objecte actual
        @param puntDePartida Posició on es començarà a llegir, prenent la posició de lectura actual com a 0
        @param llargada Nombre de valors que es vol llegir
        @return El nombre de bytes que s'ha pogut llegir
        @throws IOException La llargada de valors que es vol llegir no és múltiple de la mida de quadre
    */
    @Override
    public int read(byte[] dadesAB, int puntDePartida, int llargada) throws IOException {
        int llegit;
        if (this.sHaArribatAlFinalDelFitxer()) {
            return -1;
        }
        if (llargada % this.midaDeQuadre != 0) {
            throw new IOException("La llargada ha de ser múltiple de la mida de quadre (llargada = " + llargada + ", mida de quadre = " + this.midaDeQuadre + ")");
        }
        if (this.llargadaEnBytes != -1) {
            llargada = (int)Math.min((long)llargada, this.llargadaEnBytes - this.posicio);
        }
        int bytesLlegits = 0;
        do {
            if ((llegit = this.flux.read(dadesAB, puntDePartida, llargada)) <= 0) continue;
            bytesLlegits += llegit;
            llargada -= llegit;
            puntDePartida += llegit;
        } while (llegit > 0 && llargada > 0);
        if (bytesLlegits <= 0 && llegit == -1) {
            bytesLlegits = -1;
        } else {
            this.posicio += (long)bytesLlegits;
        }
        return bytesLlegits;
    }

    /**
        @pre ---
        @post Retorna cert si s'ha arribat al final del fitxer; altrament retorna fals.
        @return Cert si s'ha arribat al final del fitxer, altrament fals
    */
    private boolean sHaArribatAlFinalDelFitxer() {
        return this.posicio >= this.llargadaEnBytes && this.llargadaEnBytes != -1;
    }
}
```

FFT.java:

```
package cigales;

/*
 *  Copyright 2006-2007 Columbia University.
 *
 *  This file is part of MEAPsoft.
 *
 *  MEAPsoft is free software; you can redistribute it and/or modify
 *  it under the terms of the GNU General Public License version 2 as
 *  published by the Free Software Foundation.
 *
 *  MEAPsoft is distributed in the hope that it will be useful, but
 *  WITHOUT ANY WARRANTY; without even the implied warranty of
 *  MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 *  General Public License for more details.
 *
 *  You should have received a copy of the GNU General Public License
 *  along with MEAPsoft; if not, write to the Free Software
 *  Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA
 *  02110-1301 USA
 *
 *  See the file "COPYING" for the text of the license.
 */

/**
   @brief FFT és una classe que conté una implementació eficient de la FFT que he adaptat al Java. Hi incloc una advertència de la llicència del codi base encara que
    * l'hagi traduït i modificat. La utilitzo per calcular la transformada ràpida de Fourier d'un fitxer de so en poc temps.
 */


public class FFT {

 /** Guarda el nombre de valors d'amplitud de la FFT. */
 int n;

 /** Guarda el logaritme en base 2 de n. */
 int m;

 /** Guarda els valors cosinus del càlcul de la transformada de Fourier. */
 double[] cos;

 /** Guarda els valors sinus del càlcul de la transformada de Fourier. */
 double[] sin;

 /** Guarda els valors de finestra per al càlcul de la transformada de Fourier. */
 double[] finestra;


 /**
     @throws OutOfMemoryError
     @pre ---
     @post Crea un nou objecte FFT preparat per a una llista de n valors d'amplitud.
     @param n Nombre de valors de la FFT
  */
 public FFT(int n) throws OutOfMemoryError {
  this.n = n;
  this.m = (int)(Math.log(n) / Math.log(2));

  //n ha de ser una potència de 2
  if(n != (1<<m))
   throw new RuntimeException("la llargada de la FFT ha de ser una potència de 2");

  System.gc();

  cos = new double[n/2];
  sin = new double[n/2];

  for(int i=0; i<n/2; i++) {
   cos[i] = Math.cos(-2*Math.PI*i/n);
   sin[i] = Math.sin(-2*Math.PI*i/n);
  }

  finestra = new double[n];
  for(int i = 0; i < finestra.length; i++)
   finestra[i] = 0.42 - 0.5 * Math.cos(2*Math.PI*i/(n-1))
    + 0.08 * Math.cos(4*Math.PI*i/(n-1));
 }

 /*******************************************************************
 * fft.c
 * Douglas L. Jones
 * University of Illinois at Urbana-Champaign
 * January 19, 1992
 * http://cnx.rice.edu/content/m12016/latest/
 *
 *   fft: in-place radix-2 DIT DFT of a complex input
 *
 *   input:
 * n: length of FFT: must be a power of two
 * m: n = 2**m
 *   input/output
 * x: double array of length n with real part of data
 * y: double array of length n with imag part of data
 *
 *   Permission to copy and use this program is granted
 *   as long as this header is included.
 ********************************************************************/

 /**
     @throws OutOfMemoryError
```

```
     @pre ---
     @post Omple les llistes x i y amb els components reals i imaginaris, respectivament, dels valors de la FFT.
     @param x Llista on es guardarà els components reals dels valors de la transformada de Fourier
     @param y Llista on es guardarà els components imaginaris dels valors de la transformada de Fourier
   */
  public void fft(double[] x, double[] y) throws OutOfMemoryError
  {

    int i,j,k,n1,n2,a;
    double c,s,t1,t2;


    //Inverteix els bits
    j = 0;
    n2 = n/2;
    for (i=1; i < n - 1; i++) {
      n1 = n2;
      while ( j >= n1 ) {
        j = j - n1;
        n1 = n1/2;
      }
      j = j + n1;

      if (i < j) {
        t1 = x[i];
        x[i] = x[j];
        x[j] = t1;
        t1 = y[i];
        y[i] = y[j];
        y[j] = t1;
      }
    }

    // FFT
    n1 = 0;
    n2 = 1;

    for (i=0; i < m; i++) {
      n1 = n2;
      n2 = n2 + n2;
      a = 0;

      for (j=0; j < n1; j++) {
        c = cos[a];
        s = sin[a];
        a +=  1 << (m-i-1);

        for (k=j; k < n; k=k+n2) {
          t1 = c*x[k+n1] - s*y[k+n1];
          t2 = s*x[k+n1] + c*y[k+n1];
          x[k+n1] = x[k] - t1;
          y[k+n1] = y[k] - t2;
          x[k] = x[k] + t1;
          y[k] = y[k] + t2;
        }
      }
    }
  }

}
```

FormatWav.java:

```
package cigales;

import java.util.HashMap;

/**
   @brief FormatWav és una classe que conté totes les dades relatives al format d'un fitxer .wav. S'utilitza per guardar la informació relativa al format del fitxer que s'analitza.
 */

public class FormatWav {

   /** Guarda el nombre de quadres que té el fitxer WAV llegit (quadre = "frame" en anglès). */
   private int llargadaEnQuadres;

   /** Guarda les propietats del format del fitxer WAV llegit. */
   private HashMap<String, Object> propietats;

   /** Guarda la freqüència de mostreig del fitxer WAV llegit. */
   private float frequenciaDeMostreig;

   /** Guarda el nombre de canals del fitxer WAV llegit. */
   private int canals;

   /** Guarda la mida dels quadres del fitxer WAV llegit. */
   private int midaDeQuadre;

   /** Guarda el nombre de quadres per segon del fitxer WAV llegit. */
   private float quadresPerSegon;

   /** Guarda un booleà que indica si el fitxer WAV llegit està codificat en format Big Endian (cert) o en Little Endian (fals). */
   private boolean bigEndian;

   /**
      @pre ---
      @post Crea un nou objecte FormatWav amb les dades entrades per paràmetre.
      @param frequenciaMostreig Freqüència de mostreig del fitxer WAV llegit
      @param canals Nombre de canals del fitxer WAV llegit
      @param midaDeQuadre Mida dels quadres del fitxer WAV llegit
      @param quadresPerSegon Nombre de quadres per segon del fitxer WAV llegit
      @param bBigEndian Freqüència de mostreig del fitxer WAV llegit
      @param llargadaEnQuadres Booleà que indica si el fitxer WAV llegit està codificat en format Big Endian (cert) o en Little Endian (fals)
   */
   public FormatWav(float frequenciaMostreig, int canals, int midaDeQuadre, float quadresPerSegon, boolean bBigEndian, int llargadaEnQuadres) {
      this(frequenciaMostreig, canals, midaDeQuadre, quadresPerSegon, bBigEndian, null, llargadaEnQuadres);
   }

   /**
      @pre ---
      @post Crea un nou objecte FormatWav amb les dades entrades per paràmetre.
      @param frequenciaMostreig Freqüència de mostreig del fitxer WAV llegit
      @param canals Nombre de canals del fitxer WAV llegit
      @param midaDeQuadre Mida dels quadres del fitxer WAV llegit
      @param quadresPerSegon Nombre de quadres per segon del fitxer WAV llegit
      @param bBigEndian Freqüència de mostreig del fitxer WAV llegit
      @param propietats Propietats del format del fitxer WAV llegit
      @param llargadaEnQuadres Booleà que indica si el fitxer WAV llegit està codificat en format Big Endian (cert) o en Little Endian (fals)
   */
   public FormatWav(float frequenciaMostreig, int canals, int midaDeQuadre, float quadresPerSegon, boolean bBigEndian, HashMap<String, Object> propietats, int llargadaEnQuadres) {
      this.frequenciaDeMostreig = frequenciaMostreig;
      this.canals = canals;
      this.midaDeQuadre = midaDeQuadre;
      this.quadresPerSegon = quadresPerSegon;
      this.bigEndian = bBigEndian;
      this.inicialitzaMapes(propietats);
      this.llargadaEnQuadres = llargadaEnQuadres;
   }

   /**
      @pre ---
      @post Retorna el nombre de quadres que té el fitxer WAV llegit.
      @return El nombre de quadres que té el fitxer WAV llegit
   */
   public int obtinguesLlargadaEnQuadres() {
      return this.llargadaEnQuadres;
   }

   /**
      @pre ---
      @post Inicialitza els mapes de propietats sobre el format del fitxer WAV llegit.
   */
   private void inicialitzaMapes(HashMap<String, Object> properties) {
      this.propietats = new HashMap<>();
      if (properties != null) {
         this.propietats.putAll(properties);
      }
   }

   /**
      @pre ---
      @post Retorna les propietats del format del fitxer WAV llegit.
      @return Les propietats del format del fitxer WAV llegit
   */
   public HashMap<String, Object> obtinguesPropietats() {
      return this.propietats;
   }

   /**
      @pre ---
      @post Retorna la propietat del format del fitxer WAV llegit corresponent a la clau entrada per paràmetre.
      @param clau Clau a partir de la qual s'ha de cercar la propietat que es retornarà
      @return La propietat del format del fitxer WAV llegit corresponent a la clau entrada per paràmetre
   */
```

```
  public Object obtinguesPropietat(String clau) {
    return this.propietats.get(clau);
  }

  /**
    @pre ---
    @post Retorna la freqüència de mostreig del fitxer WAV llegit.
    @return La freqüència de mostreig del fitxer WAV llegit
  */
  public float obtinguesFrequenciaDeMostreig() {
    return this.frequenciaDeMostreig;
  }

  /**
    @pre ---
    @post Retorna la mida de quadre del fitxer WAV llegit.
    @return La mida de quadre del fitxer WAV llegit
  */
  public int obtinguesMidaDeQuadre() {
    return this.midaDeQuadre;
  }

  /**
    @pre ---
    @post Retorna el nombre de quadres per segon del fitxer WAV llegit.
    @return El nombre de quadres per segon del fitxer WAV llegit
  */
  public float obtinguesQuadresPerSegon() {
    return this.quadresPerSegon;
  }
}
```

IdentificadorSilencis.java:

```
package cigales;

import Parella.Pair;
import static cigales.IdentificadorSilencis.Estat.Indefinit;
import static cigales.IdentificadorSilencis.Estat.Silenci;
import static cigales.IdentificadorSilencis.Estat.Cant;
import java.util.ArrayList;

/**
  @brief IdentificadorSilencis és una classe que permet trobar els silencis que hi ha en un enregistrament a partir dels valors d'amplitud respecte del temps extrets d'aquest.
  * La utilitzo per saber a quins trams de la gravació pot haver-hi un cant de cigala i a quins no.
 */

public class IdentificadorSilencis {

  /** Guarda l'inici del silenci mentre es determina quan s'acaba.*/
  static int inici = 0;

  /** Determina la classificació del tram que s'està analitzant (silenci, cant o indefinit).*/
  static Estat estat = Indefinit;

  /** Emmagatzema la llista d'amplituds respecte del temps que defineix la gravació.*/
  static ArrayList<Double> amplituds;

  /** Guarda la freqüència de mostreig utilitzada a l'enregistrament.*/
  static int fs;

  /** Possibles estats de la variable estat*/
  public enum Estat {
    Silenci, Cant, Indefinit
  }

  /**
    @pre ---
    @post Crea un nou objecte IdentificadorSilencis amb les amplituds i la freqüència de mostreig entrades.
    @param amplituds Llista d'amplituds respecte del temps que defineix la gravació
    @param fs Freqüència de mostreig utilitzada a l'enregistrament
  */
  public IdentificadorSilencis(ArrayList<Double> amplituds, int fs) {
    this.amplituds = amplituds;
    this.fs = fs;
  }

  /**
    @pre ---
    @post Retorna la llista de silencis que hi ha a la llista d'amplituds, en forma de parelles (moment d'inici, moment final). El moment d'inici i el de final vénen indicats en forma de milisegon respecte a l'inici de la llista d'amplituds.
    @return La llista de silencis que hi ha a la llista d'amplituds, en forma de parelles (moment d'inici, moment final)
    @param llindar Llindar d'amplitud a partir del qual es considera que es tracta d'un fragment de cant i no d'un fragment de silenci
    @param finestra Finestra que s'utilitza per agrupar valors i fer comparacions d'amplitud més invariants
  */
  public static ArrayList<Pair<Double, Double> > trobaSilencisEnTemps(double llindar, int finestra) {
    return obtinguesLlistaTemporal(trobaSilencis(llindar, finestra));
  }

  /**
    @pre ---
    @post Retorna la llista de silencis que hi ha a la llista d'amplituds, en forma de parelles (moment d'inici, moment final). El moment d'inici i el de final vénen indicats en forma d'índex sobre la llista d'amplituds.
    @return La llista de silencis que hi ha a la llista d'amplituds, en forma de parelles (moment d'inici, moment final)
    @param llindar Llindar d'amplitud a partir del qual es considera que es tracta d'un fragment de cant i no d'un fragment de silenci
    @param finestra Finestra que s'utilitza per agrupar valors i fer comparacions d'amplitud més invariants
  */
  private static ArrayList<Pair<Integer, Integer> > trobaSilencis(double llindar, int finestra) {
    ArrayList<Pair<Integer, Integer> > llista = new ArrayList<>();
    int pos = 0;
    Estat e = Indefinit;

    while (pos + finestra <= amplituds.size()) {
      if (delta(pos, finestra) < llindar) e = Silenci;
      else e = Cant;
      llista.addAll(comencaOAfegeixSilenci(e, pos));
      pos += finestra;
    }

    if (delta(pos, amplituds.size()-pos) < llindar) e = Silenci;
    else e = Cant;
    llista.addAll(silenciFinal(e, pos));
    return llista;
  }

  /**
    @pre ---
    @post Retorna el valor corresponent a la diferència entre el màxim i el mínim de les amplituds compreses entre els índexs posicio i posicio+finestra.
    @return El valor corresponent a la diferència entre el màxim i el mínim de les amplituds compreses entre els índexs posicio i posicio+finestra
    @param posicio Posició, en forma d'índex sobre la llista amplituds, que indica a partir de quin punt s'ha de comprovar la diferència entre els valors màxim i mínim
    @param finestra Finestra que s'utilitza per agrupar valors i fer comparacions d'amplitud més invariants
  */
  private static double delta(int posicio, int finestra) {
    double min = Double.MAX_VALUE;
    double max = Double.MIN_VALUE;

    for (int i = posicio; i < posicio + finestra; i++) {
      if (amplituds.get(i) < min) min = amplituds.get(i);
      if (amplituds.get(i) > max) max = amplituds.get(i);
    }

    return max - min;
  }

  /**
    @pre ---
    @post Comença un silenci o bé n'afegeix un de nou, en funció de l'estat que hi hagi fins al moment (estat) i l'estat nou (e).
    @return Una llista de silencis, en forma de parelles (moment d'inici, moment final), amb una parella en cas que l'estat fins al moment (estat) sigui Silenci i l'estat nou (e) no i buida altrament.
    @param e Estat nou (Cant, Silenci o Indefinit)
    @param pos Posició, en forma d'índex sobre la llista amplituds, que indica a quin punt es dóna el canvi d'estat
  */
  private static ArrayList<Pair<Integer, Integer> > comencaOAfegeixSilenci(Estat e, int pos) {
    ArrayList<Pair<Integer, Integer> > llista = new ArrayList<>();
    if (e == Silenci) {
      if (estat != Silenci) inici = pos;
      estat = Silenci;
    }
    else {
      if (estat == Silenci) llista.add(new Pair(inici, pos));
      estat = Cant;
    }
    return llista;
  }

  /**
    @pre ---
```

```
    @post Retorna una llista de silencis, en forma de parelles (moment d'inici, moment final), amb una parella en cas que l'estat fins al moment (estat) sigui Silenci i l'estat nou (e) també.
    @return Una llista de silencis, en forma de parelles (moment d'inici, moment final), amb una parella en cas que l'estat fins al moment (estat) sigui Silenci i l'estat nou (e) també
    @param e Estat nou (Cant, Silenci o Indefinit)
    @param pos Posició, en forma d'índex sobre la llista amplituds, que indica a quin punt es dóna el canvi d'estat
  */
  private static ArrayList<Pair<Integer, Integer> > silenciFinal(Estat e, int pos) {
    ArrayList<Pair<Integer, Integer> > llista = new ArrayList<>();
    if (e == Silenci && estat == Silenci) {
      llista.add(new Pair(inici, pos));
    }
    return llista;
  }

  /**
    @pre ---
    @post Retorna la llista de silencis entrada en forma de parelles (moment d'inici, moment final) adaptant-ne els valors. El moment d'inici i el de final de la llista entrada per paràmetre vénen indicats en forma d'índex sobre la llista d'amplituds, mentre que el moment d'inici i el de final resultants vénen indicats en forma de milisegon respecte a l'inici de la llista d'amplituds.
    @return La llista de silencis entrada en forma de parelles (moment d'inici, moment final) adaptant-ne els valors
    @param llista Llista de silencis entrada en forma de parelles (moment d'inici, moment final)
  */
  private static ArrayList<Pair<Double, Double> > obtinguesLlistaTemporal(ArrayList<Pair<Integer, Integer> > llista) {
    ArrayList<Pair<Double, Double> > llistaTemps = new ArrayList<>();
    for(Pair<Integer, Integer> llista1 : llista) {
      llistaTemps.add(new Pair(1000.0 * llista1.getFirst() / (2.0*fs), 1000.0 * llista1.getSecond() / (2.0*fs)));
    };
    return llistaTemps;
  }
}
```

LectorWav.java:

```
package cigales;

import java.io.InputStream;
import java.io.IOException;
import java.io.EOFException;
import java.io.DataInputStream;

/**
  @brief LectorWav és una classe que permet llegir totes les dades, tant les corresponents al format com les d'àudio en si, d'un fitxer .wav. S'utilitza per obtenir tota la informació necessària del fitxer que s'analitza.
 */

public class LectorWav {
  /** Nombre de bytes que pot ser llegit abans que la posició marcada sigui invalidada. */
  private static final int MAXIMA_LLARGADA = 12;

  /** Guarda un valor constant que indica el valor que s'ha de llegir a la capçalera del fitxer d'àudio per confirmar que és de tipus RIFF. */
  static  final int RIFF = 1380533830;

  /** Guarda un valor constant que indica el valor que s'ha de llegir a la capçalera del fitxer d'àudio per confirmar que és de tipus WAV. */
  static  final int WAV = 1463899717;

  /** Guarda un valor constant que indica el valor que s'ha de llegir a la capçalera per saber que s'ha arribat al fragment FMT. */
  static  final int FMT = 0x666d7420;

  /** Guarda un valor constant que indica el valor que s'ha de llegir a la capçalera per saber que s'ha arribat al fragment de dades. */
  static  final int DADES = 0x64617461;

  /**
    @pre ---
    @post Crea un nou objecte LectorWav.
  */
  public LectorWav() {

  }

  /**
    @throws OutOfMemoryError
    @pre ---
    @post Retorna, en forma d'objecte DadesAudio, tota la informació, tant la corresponent al format com les dades d'àudio en si, continguda al fitxer el flux del qual s'entra per paràmetre.
    @param flux Flux d'àudio del fitxer que es vol analitzar
    @return Tota la informació, tant la corresponent al format com les dades d'àudio en si, continguda al fitxer el flux del qual s'entra per paràmetre
    @throws Exception No s'ha pogut obtenir la informació correctament.
  */
  public DadesAudio obtinguesFluxDAudio(InputStream flux) throws OutOfMemoryError, Exception {
    //Deixa el flux d'entrada a l'inici de les dades d'àudio (havent llegit la capçalera).
    FormatWav format = obtinguesFMT(flux, true);
    //El format ja està creat, o sigui que només cal llegir des dades pròpiament d'àudio.
    return new DadesAudio(flux, format);
  }

  /**
    @throws OutOfMemoryError
    @pre S'assumeix que el flux entrat està "rebobinat".
    @post Retorna, en forma d'objecte FormatWav, tota la informació corresponent al format del fitxer el flux del qual s'entra per paràmetre.
    @param flux Flux d'àudio del fitxer que es vol analitzar
    @param reinicia Indica si s'ha de reiniciar la lectura del flux.
    @return Tota la informació corresponent al format del fitxer el flux del qual s'entra per paràmetre
    @throws Exception No s'ha pogut obtenir la informació correctament.
  */
  private FormatWav obtinguesFMT(InputStream flux, boolean reinicia) throws OutOfMemoryError, Exception {
    int nLlegits = 0;
    int fmt, llargada, tipusDeWav;
    short canals;
    long frequenciaMostreig;
    long mitjanaBytesPerSegon;
    short alineacioDeBlocs;
    int midaDeMostraEnBits;
    int llargadaTotal;

    DataInputStream dis = new DataInputStream(flux);

    if (reinicia) {
      dis.mark(MAXIMA_LLARGADA);
    }

    int riff = dis.readInt();
    int llargadaFitxer = llegeixLong(dis);
    int wav = dis.readInt();
    if (llargadaFitxer <= 0) {
      llargadaFitxer = -1;
      llargadaTotal = -1;
    } else {
      llargadaTotal = llargadaFitxer + 8;
    }
    if ((riff != RIFF) || (wav != WAV)) {
      //No és un fitxer .wav.
      if (reinicia) {
        dis.reset();
      }
      throw new Exception("El fitxer no té format WAV.");
    }

    //El bucle continua fins que es troba l'FMT o bé el final del fitxer.
    while(true) {
      try {
        fmt = dis.readInt();
        nLlegits += 4;
        if( fmt==FMT ) {
          //S'ha trobat l'FMT.
          break;
        } else {
          //No fa cas d'aquest fragment i continua llegint.
          llargada = llegeixLong(dis);
          nLlegits += 4;
          if (llargada % 2 > 0) llargada++;
          nLlegits += dis.skipBytes(llargada);
        }
      } catch (EOFException eof) {
        //S'ha arribat al final del fitxer sense trobar el fragment FMT.
        throw new Exception("No és un fitxer WAV vàlid.");
      }
    }
    //Llegeix la llargada del fragment corresponent al format.
    llargada = llegeixLong(dis);
    nLlegits += 4;
    //Posició nLlegits després del fragment corresponent al format.
    int llargadaFinal = nLlegits + llargada;
    //Llegeix la informació referent al tipus d'enregistrament WAV de què es tracta.
    tipusDeWav = llegeixShort(dis); nLlegits += 2;

    //1 canal = monofònic, 2 canals = estereofònic
    canals = llegeixShort(dis); nLlegits += 2;
```

```
    frequenciaMostreig = llegeixLong(dis); nLlegits += 4;
    mitjanaBytesPerSegon = llegeixLong(dis); nLlegits += 4;
    alineacioDeBlocs = llegeixShort(dis); nLlegits += 2;

    //Valor específic de PCM
    midaDeMostraEnBits = (int)llegeixShort(dis); nLlegits += 2;

    //Si midaDeMostraEnBits == 8, s'ha d'utilitzar PCM_UNSIGNED.
    //Cal saltar un nombre de valors equivalent a la diferència entre la llargada de la capçalera corresponent al format i la posició que està llegint.
    //Si la llargada del fragment és senar, és que hi ha un byte afegit al final.
    if (llargada % 2 != 0) llargada += 1;
    if (llargadaFinal > nLlegits)
      nLlegits += dis.skipBytes(llargadaFinal - nLlegits);

    //Ja es té el format, per tant només queda avançar fins a trobar l'inici de les dades.
    nLlegits = 0;
    while(true) {
      try{
        int dades = dis.readInt();
        nLlegits+=4;
        if (dades == DADES) {
          //Ja ha trobat el fragment de dades.
          break;
        } else {
          //No fa cas d'aquest fragment i continua llegint.
          int aquestaLlargada = llegeixLong(dis); nLlegits += 4;
          if (aquestaLlargada % 2 > 0) aquestaLlargada++;
          nLlegits += dis.skipBytes(aquestaLlargada);
        }
      } catch (EOFException eof) {
        //S'ha arribat al final del fitxer sense trobar el fragment FMT.
        throw new Exception("No és un fitxer WAV vàlid.");
      }
    }

    //Llargada del fragment de dades
    int llargadaDades = llegeixLong(dis); nLlegits += 4;

    //Quadre = "Frame" en anglès
    int midaQuadre = calculaMidaDeQuadrePCM(midaDeMostraEnBits, canals);

    //Crea un objecte FormatWav a partir de les dades llegides.
    return new FormatWav((float)frequenciaMostreig,
                         canals,
                         midaQuadre,
                         (float)frequenciaMostreig, false,
                    llargadaDades / midaQuadre);
  }

  /**
    @pre ---
    @post Llegeix quatre bytes del flux de dades entrat per paràmetre i els retorna en forma d'enter.
    @param dis Flux de dades del fitxer que es vol analitzar
    @return Els quatre primers bytes del flux de dades entrat per paràmetre, des del punt de lectura en què aquest es troba, en forma d'enter
    @throws IOException No s'ha pogut llegir la informació correctament.
  */
  private int llegeixLong(DataInputStream dis) throws IOException {

    int b1, b2, b3, b4, i;

    i = dis.readInt();

    b1 = ( i & 0xFF ) << 24 ;
    b2 = ( i & 0xFF00 ) << 8;
    b3 = ( i & 0xFF0000 ) >> 8;
    b4 = ( i & 0xFF000000 ) >>> 24;

    i = ( b1 | b2 | b3 | b4 );

    return i;
  }

  /**
    @pre ---
    @post Llegeix dos bytes del flux de dades entrat per paràmetre i els retorna en forma d'enter.
    @param dis Flux de dades del fitxer que es vol analitzar
    @return Els dos primers bytes del flux de dades entrat per paràmetre, des del punt de lectura en què aquest es troba, en forma d'enter
    @throws IOException No s'ha pogut llegir la informació correctament.
  */
  private short llegeixShort(DataInputStream dis)  throws IOException {

    short s, high, low;

    s = dis.readShort();

    high = (short)(( s & 0xFF ) << 8) ;
    low = (short)(( s & 0xFF00 ) >>> 8);

    s = (short)( high | low );

    return s;
  }

  /**
    @pre ---
    @post Calcula i retorna la mida de quadre del fitxer que s'analitza.
    @param midaMostreigEnBits Mida de mostreig, mesurada en bits, del fitxer que s'analitza
    @param canals Nombre de canals del fitxer que s'analitza
    @return La mida de quadre del fitxer que s'analitza
  */
  private static int calculaMidaDeQuadrePCM(int midaMostreigEnBits, int canals) {
    return ((midaMostreigEnBits + 7) / 8) * canals;
  }
}
```

Parella.java:

```
package parella;


public class Parella<A, B> implements Comparable {
  private A primer;
  private B segon;

  public Parella(A primer, B segon) {
        super();
        this.primer = primer;
        this.segon = segon;
  }

  public int hashCode() {
        int hashPrimer = primer != null ? primer.hashCode() : 0;
        int hashSegon = segon != null ? segon.hashCode() : 0;

        return (hashPrimer + hashSegon) * hashSegon + hashPrimer;
  }

  public boolean equals(Object altra) {
        if (altra instanceof Parella) {
              Parella altraParella = (Parella) altra;
              return
              ((  this.primer == altraParella.primer ||
                    ( this.primer != null && altraParella.primer != null &&
                      this.primer.equals(altraParella.primer))) &&
               (    this.segon == altraParella.segon ||
                    ( this.segon != null && altraParella.segon != null &&
                      this.segon.equals(altraParella.segon))) );
        }

        return false;
  }

  public String toString()
  {
      return "(" + primer + ", " + segon + ")";
  }

  public A getPrimer() {
        return primer;
  }

  public void setPrimer(A primer) {
        this.primer = primer;
  }

  public B getSegon() {
        return segon;
  }

  public void setSegon(B segon) {
        this.segon = segon;
  }

  @Override
  public int compareTo(Object altra) {
    if (altra instanceof Parella) {
              Parella altraParella = (Parella) altra;
          if (altraParella.primer instanceof Double && altraParella.segon instanceof Double) {
```

```
            if (((Double) altraParella.primer).compareTo(((Double) this.primer)) > 0 ||
((Double) altraParella.segon).compareTo(((Double) this.segon)) > 0) return 1;
            else if (((Double) altraParella.primer).compareTo(((Double) this.primer)) < 0
|| ((Double) altraParella.segon).compareTo(((Double) this.segon)) < 0) return -1;
            else return 0;
          }
          else return -1;
              }
    else return -1;
  }
}
```

Aplicació per a Android:

ActivitatPrincipal.java:

```
package cigales.cicadaapp;

import android.app.AlertDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.res.Resources;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.util.DisplayMetrics;
import android.view.LayoutInflater;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.RelativeLayout;

import java.util.Locale;

/**
 ActivitatPrincipal és la classe encarregada de gestionar la pantalla principal de
l'aplicació.
 */

public class ActivitatPrincipal extends AppCompatActivity implements
View.OnClickListener {

  Button botoLlistaEspecies;
  Button botoGravacio;
  Button botoAnalisi;
  Context context = this;
  Configuracio configuracio = new Configuracio(context);

  ListView llistaIdiomes;
  String[] nomsIdiomes ={
      "Català",
      "Castellano",
      "English",
      "Français"
  };

  Integer[] imatgesIdiomes={
      R.drawable.catala_mini,
      R.drawable.castella_mini,
      R.drawable.angles_mini,
      R.drawable.frances_mini
  };

  ListView llistaConfiguracio;
  String[] nomsConfiguracio ={
      "Instruccions d'ús",
      "Quant a"
  };

  RelativeLayout disseny;

  AlertDialog dialeg;

  private String[] obtinguesNomsConfiguracioActualitzats() {
    String[] s = {getResources().getString(R.string.Instruccions),
getResources().getString(R.string.QuantA)};
    return s;
  }

  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    //Obre l'aplicació amb l'idioma amb què es va tancar l'últim cop, o en català si és
el primer cop que s'obre.
    String[] parametresLlegits = configuracio.llegeix(3, "configuració.txt");

    String codiIdioma = "";

    if (parametresLlegits[0].compareTo("Català") == 0) {
      codiIdioma = "cat";
    }
    else if (parametresLlegits[0].compareTo("Castellano") == 0) {
      codiIdioma = "es";
    }
    else if (parametresLlegits[0].compareTo("English") == 0) {
      codiIdioma = "eng";
    }
    else if (parametresLlegits[0].compareTo("Français") == 0) {
      codiIdioma = "fr";
    }

    Resources res = context.getResources();
    DisplayMetrics dm = res.getDisplayMetrics();
    android.content.res.Configuration conf = res.getConfiguration();
    conf.setLocale(new Locale(codiIdioma));
    res.updateConfiguration(conf, dm);

    setContentView(R.layout.activity_activitat_principal);


    //Llista d'espècies
    botoLlistaEspecies = (Button)findViewById(R.id.LlistaEspecies);
    botoLlistaEspecies.setOnClickListener(this);

    //Gravació
    botoGravacio = (Button)findViewById(R.id.Gravacio);
    botoGravacio.setOnClickListener(this);

    //Anàlisi de fitxers
    botoAnalisi = (Button)findViewById(R.id.AnalisiFitxer);
    botoAnalisi.setOnClickListener(this);

    //Llista d'idiomes
    AdaptadorListViewIdiomes adaptadorIdiomes=new
AdaptadorListViewIdiomes(this, nomsIdiomes, imatgesIdiomes);
    llistaIdiomes=(ListView)findViewById(R.id.LlistaIdiomes);
    llistaIdiomes.setAdapter(adaptadorIdiomes);

    //Llista de configuració
    AdaptadorListViewConfiguracio adaptadorConfiguracio=new
AdaptadorListViewConfiguracio(this, nomsConfiguracio);
    llistaConfiguracio=(ListView)findViewById(R.id.LlistaConfiguracio);
    llistaConfiguracio.setAdapter(adaptadorConfiguracio);

    //Disseny general de la pantalla
    disseny = (RelativeLayout) findViewById(R.id.disseny);

    llistaIdiomes.setOnItemClickListener(new AdapterView.OnItemClickListener() {

      @Override
      public void onItemClick(AdapterView<?> pare, View vista,
                  int posicio, long id) {
        String idiomaSeleccionat = nomsIdiomes[+posicio];

        canviaIdioma(idiomaSeleccionat, true);

        llistaIdiomes.setVisibility(View.GONE);
        llistaConfiguracio.setVisibility(View.GONE);
      }
    });

    llistaConfiguracio.setOnItemClickListener(new
AdapterView.OnItemClickListener() {

      @Override
      public void onItemClick(AdapterView<?> pare, View vista,
                  int posicio, long id) {

        if (posicio == 0) mostraInstruccions();
        else if (posicio == 1) mostraQuantA();

        llistaIdiomes.setVisibility(View.GONE);
        llistaConfiguracio.setVisibility(View.GONE);
      }
    });

    disseny.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        llistaIdiomes.setVisibility(View.GONE);
```

```
            llistaConfiguracio.setVisibility(View.GONE);
        }
    });

    canviaIdioma(parametresLlegits[0], false);
}

private void mostraQuantA() {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    LayoutInflater inflater = this.getLayoutInflater();

    builder.setView(inflater.inflate(R.layout.quant_a, null))
        .setNeutralButton(R.string.DAcord, new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int id) {
                if (dialeg != null) dialeg.cancel();
            }
        });
    dialeg = builder.create();
    dialeg.show();
}

private void mostraInstruccions() {
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    LayoutInflater inflater = this.getLayoutInflater();

    builder.setView(inflater.inflate(R.layout.instruccions, null))
        .setNeutralButton(R.string.DAcord, new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int id) {
                if (dialeg != null) dialeg.cancel();
            }
        });
    dialeg = builder.create();
    dialeg.show();

    String[] parametresLlegits = configuracio.llegeix(3, "configuració.txt");
    if (parametresLlegits.length > 2 && parametresLlegits[2].compareTo("Instruccions
mostrades") != 0) {
        parametresLlegits[2] = "Instruccions mostrades";
        configuracio.escriu(parametresLlegits, "configuració.txt");
    }
}

private void canviaIdioma(String idioma, boolean aplicaCanvis) {

    //Canvia l'idioma que apareix a la pantalla principal.
    Button botoIdioma=(Button)findViewById(R.id.Idioma);

    String codiIdioma = "";

    if (idioma.compareTo("Català") == 0) {
        botoIdioma.setText("Català");

botoIdioma.setCompoundDrawablesWithIntrinsicBounds(R.drawable.catala_mini_es
pai, 0, 0, 0);
        codiIdioma = "cat";
    }
    else if (idioma.compareTo("Castellano") == 0) {
        botoIdioma.setText("Castellano");

botoIdioma.setCompoundDrawablesWithIntrinsicBounds(R.drawable.castella_mini_e
spai, 0, 0, 0);
        codiIdioma = "es";
    }
    else if (idioma.compareTo("English") == 0) {
        botoIdioma.setText("English");

botoIdioma.setCompoundDrawablesWithIntrinsicBounds(R.drawable.angles_mini_es
pai, 0, 0, 0);
        codiIdioma = "eng";
    }
    else if (idioma.compareTo("Français") == 0) {
        botoIdioma.setText("Français");

botoIdioma.setCompoundDrawablesWithIntrinsicBounds(R.drawable.frances_mini_e
spai, 0, 0, 0);
        codiIdioma = "fr";
    }

    Resources res = context.getResources();
    DisplayMetrics dm = res.getDisplayMetrics();
    android.content.res.Configuration conf = res.getConfiguration();
    conf.setLocale(new Locale(codiIdioma));
    res.updateConfiguration(conf, dm);

    //Aplica el canvi al fitxer configuració.txt.
    if (aplicaCanvis) {
        String[] parametresLlegits = configuracio.llegeix(3, "configuració.txt");

        if (parametresLlegits[0].compareTo(idioma) != 0) {
            parametresLlegits[0] = idioma;
            configuracio.escriu(parametresLlegits, "configuració.txt");
        }

        finish();
        overridePendingTransition(0, 0);

this.startActivity(getIntent().addFlags(Intent.FLAG_ACTIVITY_NO_ANIMATION));
        overridePendingTransition(0, 0);
    }
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    return super.onOptionsItemSelected(item);
}

@Override
public void onClick(View view) {
    switch (view.getId()) {
        case R.id.LlistaEspecies:
            botoLlistaEspeciesClic();
            break;
        case R.id.Gravacio:
            botoGravacioClic();
            break;
        case R.id.AnalisiFitxer:
            botoAnalisiFitxerClic();
            break;
    }
}

private void botoLlistaEspeciesClic() {
    llistaIdiomes.setVisibility(View.GONE);
    llistaConfiguracio.setVisibility(View.GONE);

    startActivity(new Intent("cigales.cicadaapp.LlistaEspecies"));
}

private void botoGravacioClic() {
    llistaIdiomes.setVisibility(View.GONE);
    llistaConfiguracio.setVisibility(View.GONE);

    String[] parametresLlegits = configuracio.llegeix(3, "configuració.txt");
    if (parametresLlegits.length > 2 && parametresLlegits[2].compareTo("Instruccions
mostrades") != 0) {
        mostraInstruccions();
        parametresLlegits[2] = "Instruccions mostrades";
        configuracio.escriu(parametresLlegits, "configuració.txt");
    }
    else startActivity(new Intent("cigales.cicadaapp.EnregistradorAudio"));
}

private void botoAnalisiFitxerClic() {
    llistaIdiomes.setVisibility(View.GONE);
    llistaConfiguracio.setVisibility(View.GONE);

    startActivity(new Intent("cigales.cicadaapp.ExploradorFitxers"));
}


public void mostraIdiomes(View view) {
    llistaIdiomes.setVisibility(View.VISIBLE);
    llistaConfiguracio.setVisibility(View.GONE);
}

public void mostraMenuConfiguracio(View view) {
    nomsConfiguracio = obtinguesNomsConfiguracioActualitzats();
    AdaptadorListViewConfiguracio adaptadorConfiguracio=new
AdaptadorListViewConfiguracio(this, nomsConfiguracio);
    llistaConfiguracio=(ListView)findViewById(R.id.LlistaConfiguracio);
    llistaConfiguracio.setAdapter(adaptadorConfiguracio);
    llistaConfiguracio.setVisibility(View.VISIBLE);
    llistaIdiomes.setVisibility(View.GONE);
}
}
```

AdaptadorListViewConfiguracio.java:

```
package cigales.cicadaapp;

import android.app.Activity;
import android.view.LayoutInflater;
```

```
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;

/**
 AdaptadorListViewConfiguracio és la classe encarregada de gestionar la llista
d'opcions de configuració que es mostra a la pantalla principal de l'aplicació. Aquestes
opcions permeten visualitzar la informació referent a l'aplicació o bé les seves
instruccions d'ús.
 */

public class AdaptadorListViewConfiguracio extends ArrayAdapter<String> {

   private final Activity context;
   private final String[] text;

   public AdaptadorListViewConfiguracio(Activity context, String[] text) {
      super(context, R.layout.llista_configuracio, text);

      this.context=context;
      this.text=text;
   }

   public View getView(int posicio,View vista_,ViewGroup pare) {
      LayoutInflater inflador=context.getLayoutInflater();
      View vista=inflador.inflate(R.layout.llista_configuracio, null, true);

      TextView vistaText = (TextView) vista.findViewById(R.id.item);

      vistaText.setText(text[posicio]);
      return vista;

   };
}
```

AdaptadorListViewFitxers.java:

```
package cigales.cicadaapp;

import java.util.List;
import android.content.Context;
import android.graphics.drawable.Drawable;
import android.support.v4.content.ContextCompat;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

/**
 AdaptadorListViewFitxers és la classe encarregada de gestionar la llista de fitxers i
directoris que es mostra en voler analitzar un fitxer de so enregistrat amb antelació.
 */

public class AdaptadorListViewFitxers extends ArrayAdapter<Fitxer>{

   private Context c;
   private int id;
   private List<Fitxer> fitxers;

   public AdaptadorListViewFitxers(Context context, int idTextView,
                           List<Fitxer> objectes) {
      super(context, idTextView, objectes);
      c = context;
      id = idTextView;
      fitxers = objectes;
   }

   @Override
   public Fitxer getItem(int i)
   {
      return fitxers.get(i);
   }

   @Override
   public View getView(int posicio, View vista, ViewGroup carpetaContenidora) {
      View v = vista;
      if (v == null) {
         LayoutInflater                                vi                                =
(LayoutInflater)c.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
         v = vi.inflate(id, null);
      }

      final Fitxer f = fitxers.get(posicio);
      if (f != null) {
         TextView t1 = (TextView) v.findViewById(R.id.Text1);
         TextView t2 = (TextView) v.findViewById(R.id.Text2);
         TextView t3 = (TextView) v.findViewById(R.id.Data);

         //Assigna la imatge corresponent (carpeta, fitxer no vàlid o fitxer vàlid)
         ImageView vistaImatge = (ImageView) v.findViewById(R.id.Icona);
         String uri = "drawable/" + f.getImatge();
         int idImatge = c.getResources().getIdentifier(uri, null, c.getPackageName());
         Drawable imatge = ContextCompat.getDrawable(c, idImatge);
         vistaImatge.setImageDrawable(imatge);

         if(t1!=null)
            t1.setText(f.getNom());
         if(t2!=null)
            t2.setText(f.getInformacio());
         if(t3!=null)
            t3.setText(f.getData());
      }
      return v;
   }
}
```

AdaptadorListViewIdiomes.java:

```
package cigales.cicadaapp;

import android.app.Activity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;

/**
 AdaptadorListViewIdiomes és la classe encarregada de gestionar la llista d'idiomes que
es mostra a la pantalla principal de l'aplicació.
 */

public class AdaptadorListViewIdiomes extends ArrayAdapter<String> {

   private final Activity context;
   private final String[] text;
   private final Integer[] imatge;

   public AdaptadorListViewIdiomes(Activity context, String[] text, Integer[] imatge) {
      super(context, R.layout.llista_idiomes, text);

      this.context=context;
      this.text=text;
      this.imatge=imatge;
   }

   public View getView(int posicio,View vista_,ViewGroup pare) {
      LayoutInflater inflador=context.getLayoutInflater();
      View vista=inflador.inflate(R.layout.llista_idiomes, null, true);

      TextView vistaText = (TextView) vista.findViewById(R.id.item);
      ImageView vistaImatge = (ImageView) vista.findViewById(R.id.icon);

      vistaText.setText(text[posicio]);
      vistaImatge.setImageResource(imatge[posicio]);
      return vista;
   }
}
```

Configuracio.java:

```
package cigales.cicadaapp;


import android.content.Context;
import android.content.ContextWrapper;
import android.util.Log;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;

/**
 Configuracio és la classe que s'utilitza per llegir i escriure les opcions de configuració
actuals: idioma, directori des d'on s'ha d'analitzar els fitxers de so i indicador de si les
instruccions d'ús de l'aplicació han estat visualitzades per l'usuari o no.
```

```
 */

public class Configuracio {

   Context context;

   public Configuracio() {
   }

   public Configuracio(Context c) {
      context = c;
   }

   public String[] llegeix(int nCamps, String nomFitxer) {

      String[] texts = new String[nCamps];

      try {
         InputStream entrada = new
ContextWrapper(context).openFileInput(nomFitxer);

         if ( entrada != null ) {
            InputStreamReader lector = new InputStreamReader(entrada);
            BufferedReader lectorAmbBuffer = new BufferedReader(lector);
            String text;

            int i = 0;

            while ((text = lectorAmbBuffer.readLine()) != null && i < nCamps) {
               texts[i] = text;
               i++;
            }

            entrada.close();

            if (i < nCamps) {
               String[] informacions = {"Català", "/sdcard/", "Instruccions no mostrades"};
               escriu(informacions, "configuració.txt");
               texts = llegeix(nCamps, nomFitxer);
            }
         }
      }
      catch (FileNotFoundException e) {
         Log.e("Error", "Fitxer no trobat: " + e.toString());
         String[] informacions = {"Català", "/sdcard/", "Instruccions no mostrades"};
         escriu(informacions, "configuració.txt");
         texts = llegeix(nCamps, nomFitxer);
      } catch (IOException e) {
         Log.e("Error", "No es pot llegir el fitxer: " + e.toString());
      }

      return texts;
   }

   public void escriu(String[] informacions, String nomFitxer) {
      try {
         FileOutputStream escriptor = new
ContextWrapper(context).openFileOutput(nomFitxer, Context.MODE_PRIVATE);
         escriptor.write((informacions[0] + "\n" + informacions[1] + "\n" +
informacions[2] + "\n").getBytes());
         escriptor.close();
      }
      catch (IOException e) {
         Log.e("Error", "No es pot escriure al fitxer: " + e.toString());
      }
   }
}
```

EnregistradorAudio.java:

```
package cigales.cicadaapp;

import android.content.Intent;
import android.os.CountDownTimer;
import android.os.Handler;
import android.os.Looper;
import android.os.Message;
import android.support.v7.app.AppCompatActivity;
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.ImageView;
import android.os.Bundle;
import android.os.Environment;
import android.widget.Button;
import android.view.View;
import android.content.Context;
import android.util.Log;
import android.media.MediaRecorder;
import android.media.MediaPlayer;
import android.widget.TextView;
import android.widget.Toast;

import java.io.IOException;
import java.util.ArrayList;
import java.util.TimerTask;
import java.util.concurrent.Executors;
import java.util.concurrent.ScheduledExecutorService;
import java.util.concurrent.TimeUnit;

import cigales.Cigales;

/**
 EnregistradorAudio és la classe que gestiona la pantalla d'enregistrament, reproducció
i anàlisi dels fitxers de so.
 */

public class EnregistradorAudio extends AppCompatActivity implements
View.OnClickListener
{
   private static final String LOG_TAG = "Gravació en curs";
   private static String nomFitxer = null;

   private MediaRecorder enregistrador = null;

   private MediaPlayer   reproductor = null;

   private ImageView microfon;
   private ImageView microfonFonsBlanc;
   private ImageView rellotgeDeSorra;
   private ImageView carregant;
   private ImageView reproduccio;
   private TextView gravacioEnCurs;
   private TextView reproduccioEnCurs;
   private TextView analitzant;
   private Button botoReprodueix = null;
   private Button botoAnalitza = null;
   private Button botoDescarta = null;
   private Button botoAtura = null;

   Animation rotacio;
   Animation parpalleig;

   public Handler mHandlerResultatsObtinguts = new Handler() {
      public void handleMessage(Message msg) {
         botoAnalitza2Clic();
      }
   };

   public Handler mHandlerErrorMemoria = new Handler() {
      public void handleMessage(Message msg) {
         final Toast missatge = Toast.makeText(getApplicationContext(),
context.getResources().getString(R.string.MemoriaInsuficient),
               Toast.LENGTH_LONG);

         missatge.show();

         new CountDownTimer(9000, 1000)
         {
            public void onTick(long milisegonsQueQueden) {missatge.show();}
            public void onFinish() {
            }

         }.start();

         acabaAnalisi();
      }
   };

   public Handler mHandlerErrorAnalisi = new Handler() {
      public void handleMessage(Message msg) {
         final Toast missatge = Toast.makeText(getApplicationContext(),
context.getResources().getString(R.string.ErrorAnalisi),
               Toast.LENGTH_LONG);

         missatge.show();

         new CountDownTimer(9000, 1000)
         {
            public void onTick(long milisegonsQueQueden) {missatge.show();}
            public void onFinish() {
            }

         }.start();

         acabaAnalisi();
      }
   };

   String resultatsIdentificacio = "";
```

```
String auxiliar;
boolean errorMemoria;
boolean errorAnalisi;

boolean analisiFitxer = false;

Context context = this;

Cigales cigales;

private EnregistradorWav enregistradorWav = new EnregistradorWav();

@Override
public void onCreate(Bundle icicle) {
  super.onCreate(icicle);

  setContentView(R.layout.activity_enregistrador_audio);

  rotacio = AnimationUtils.loadAnimation(getApplicationContext(),
      R.anim.rotacio);

  parpalleig = AnimationUtils.loadAnimation(getApplicationContext(),
      R.anim.parpalleig);

  //Micròfon
  microfon = (ImageView)findViewById(R.id.Microfon);

  //Micròfon de fons blanc
  microfonFonsBlanc = (ImageView)findViewById(R.id.MicrofonFonsBlanc);

  //Rellotge de sorra
  rellotgeDeSorra = (ImageView)findViewById(R.id.RellotgeSorra);

  //Roda
  carregant = (ImageView)findViewById(R.id.Carregant);

  //Reproducció
  reproduccio = (ImageView)findViewById(R.id.Reproduccio);

  //Gravació en curs
  gravacioEnCurs = (TextView)findViewById(R.id.GravacioEnCurs);

  //Reproducció en curs
  reproduccioEnCurs = (TextView)findViewById(R.id.ReproduccioEnCurs);

  //Analitzant
  analitzant = (TextView)findViewById(R.id.Analitzant);

  //Botó reprodueix
  botoReprodueix = (Button)findViewById(R.id.Reprodueix);

  //Botó analitza
  botoAnalitza = (Button)findViewById(R.id.Analitza);

  //Botó descarta
  botoDescarta = (Button)findViewById(R.id.Descarta);

  //Botó atura
  botoAtura = (Button)findViewById(R.id.Atura);

  microfon.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
      microfonClic();
    }
  });

  microfonFonsBlanc.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
      microfonFonsBlancClic();
    }
  });

  botoReprodueix.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
      botoReprodueixClic();
    }
  });


  //S'ha de cridar fils d'execució diferents, altrament els canvis a la interfície posteriors a haver obtingut els resultats de l'anàlisi no són aplicats.
  botoAnalitza.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
      botoAnalitzaClic();

      errorMemoria = false;
      errorAnalisi = false;
      auxiliar = resultatsIdentificacio;

      //Quan es detecta que hi ha hagut un canvi a resultatsIdentificacio, errorMemoria o errorAnalisi, s'aplica els canvis a la interfície i/o es mostra els missatges corresponents. No es pot fer canvis a la interfície des d'un fil diferent, per això s'ha d'utilitzar aquest mètode.
      Thread analitzador = new Thread(new Runnable() {
        @Override
        public void run() {
          Looper.prepare();
          try {
            cigales = new Cigales();
            resultatsIdentificacio = cigales.identificaCant(nomFitxer);
          }
          catch (OutOfMemoryError e) {
            errorMemoria = true;
          }
          catch (Exception e) {
            errorAnalisi = true;
          }
        }
      },"Fil Analitzador");

      analitzador.start();

      iniciaTemporitzadorResultats();
      iniciaTemporitzadorErrorMemoria();
      iniciaTemporitzadorErrorAnalisi();
    }
  });

  botoDescarta.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
      botoDescartaClic();
    }
  });

  botoAtura.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
      botoAturaClic();
    }
  });

  Bundle b = getIntent().getExtras();
  if(b != null) {
    nomFitxer = b.getString("fitxerPerAnalitzar");
    analisiFitxer = true;
    botoAnalitza.callOnClick();
  }
  else analisiFitxer = false;
}

protected void iniciaTemporitzadorResultats() {
  ScheduledExecutorService temporitzador = Executors.newScheduledThreadPool(1);
  temporitzador.schedule(new TimerTask() {
    public void run() {
      while (auxiliar.compareTo(resultatsIdentificacio) == 0) {
        try {
          Thread.sleep(250);
        } catch (InterruptedException e) {
          e.printStackTrace();
        }
      }
      mHandlerResultatsObtinguts.obtainMessage(1).sendToTarget();
    }
  }, (long) 10, TimeUnit.MILLISECONDS);
  temporitzador.shutdown();
}

protected void iniciaTemporitzadorErrorMemoria() {
  ScheduledExecutorService temporitzador = Executors.newScheduledThreadPool(1);
  temporitzador.schedule(new TimerTask() {
    public void run() {
      while (!errorMemoria) {
        try {
          Thread.sleep(250);
        } catch (InterruptedException e) {
          e.printStackTrace();
        }
      }
      mHandlerErrorMemoria.obtainMessage(1).sendToTarget();
    }
  }, (long) 10, TimeUnit.MILLISECONDS);
  temporitzador.shutdown();
}
```

```
    protected void iniciaTemporitzadorErrorAnalisi() {
        ScheduledExecutorService temporitzador =
Executors.newScheduledThreadPool(1);
        temporitzador.schedule(new TimerTask() {
            public void run() {
                while (!errorAnalisi) {
                    try {
                        Thread.sleep(250);
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }
                }
                mHandlerErrorAnalisi.obtainMessage(1).sendToTarget();
            }
        }, (long) 10, TimeUnit.MILLISECONDS);
        temporitzador.shutdown();
    }

    private void botoDescartaClic() {
        botoReprodueix.setVisibility(View.GONE);
        botoAnalitza.setVisibility(View.GONE);
        botoDescarta.setVisibility(View.GONE);
        rellotgeDeSorra.setVisibility(View.GONE);
        microfonFonsBlanc.setEnabled(true);
    }

    private void botoAnalitzaClic() {
        botoReprodueix.setVisibility(View.GONE);
        botoAnalitza.setVisibility(View.GONE);
        botoDescarta.setVisibility(View.GONE);
        microfonFonsBlanc.setVisibility(View.GONE);
        rellotgeDeSorra.setVisibility(View.VISIBLE);
        analitzant.setVisibility(View.VISIBLE);
        analitzant.setAnimation(parpalleig);
    }

    private void botoAnalitza2Clic() {
        try {
            if (!(resultatsIdentificacio.substring(0,14).compareTo("No s'ha trobat") == 0 ||
resultatsIdentificacio.substring(0,14).compareTo("El cant enregi") == 0 ||
resultatsIdentificacio.substring(0,5).compareTo("Atesa") == 0 ||
resultatsIdentificacio.substring(0, 7).compareTo("Cantant") == 0)) {
                ArrayList<String> especies = emmagatzemaResultats(resultatsIdentificacio);

                Intent intent = new Intent("cigales.cicadaapp.Resultats");
                Bundle b = new Bundle();
                b.putStringArrayList("llistaEspecies", especies);
                intent.putExtras(b);
                startActivity(intent);
                auxiliar = "";
                resultatsIdentificacio = "";
                if (analisiFitxer) finish();
                else acabaAnalisi();
            }
            else {
                //Aquesta tècnica permet mostrar el missatge durant més segons, en lloc dels
predeterminats 2 o 3.
                final Toast missatge;
                int milisegonsFinals;
                if (resultatsIdentificacio.substring(0,14).compareTo("No s'ha trobat") == 0) {
                    missatge = Toast.makeText(getApplicationContext(),
context.getString(R.string.SenseCoincidencies),
                            Toast.LENGTH_LONG);
                    milisegonsFinals = 5000;
                }
                else if (resultatsIdentificacio.substring(0, 14).compareTo("El cant enregi") ==
0) {
                    missatge = Toast.makeText(getApplicationContext(),
context.getString(R.string.CantFluix),
                            Toast.LENGTH_LONG);
                    milisegonsFinals = 5000;
                }
                else if (resultatsIdentificacio.substring(0, 5).compareTo("Atesa") == 0) {
                    missatge = Toast.makeText(getApplicationContext(),
context.getString(R.string.MemoriaInsuficient),
                            Toast.LENGTH_LONG);
                    milisegonsFinals = 1000;
                }
                else if (resultatsIdentificacio.substring(0, 7).compareTo("Cantant") == 0) {
                    missatge = Toast.makeText(getApplicationContext(),
context.getString(R.string.CicadaOrniDeFons),
                            Toast.LENGTH_LONG);
                    milisegonsFinals = 1000;
                }
                else {
                    missatge = null;
                    milisegonsFinals = 0;
                }

                missatge.show();
                new CountDownTimer(9000, milisegonsFinals) {
                    public void onTick(long milisegonsQueQueden) {
                        missatge.show();
                    }
                    public void onFinish() {
                    }
                }.start();

                auxiliar = "";
                resultatsIdentificacio = "";

                if (analisiFitxer) {
                    finish();
                    startActivity(new Intent("cigales.cicadaapp.ExploradorFitxers"));
                }
                else acabaAnalisi();
            }
        } catch (Exception e) {
            e.printStackTrace();
            Log.e("Error", e.getMessage().toString());
        }
    }

    private void acabaAnalisi() {
        analitzant.clearAnimation();
        analitzant.setVisibility(View.GONE);
        rellotgeDeSorra.setVisibility(View.GONE);
        microfonFonsBlanc.setVisibility(View.VISIBLE);
        microfonFonsBlanc.setEnabled(true);
    }

    private ArrayList<String> emmagatzemaResultats(String resultatsIdentificacio) {
        ArrayList<String> llista = new ArrayList<>();
        boolean continua = true;
        int digitsPercentatge;
        while (continua && resultatsIdentificacio.compareTo("") != 0) {
            digitsPercentatge = 0;
            if (resultatsIdentificacio.length() > 25 && resultatsIdentificacio.substring(0,
25).compareTo("Cicada barbara lusitanica") == 0) {
                while(resultatsIdentificacio.charAt(27+digitsPercentatge) != '%')
digitsPercentatge++;
                llista.add((resultatsIdentificacio.charAt(25) != '*' ? "Cicada barbara lusitanica"
: "Cicada barbara lusitanica*") + " " + resultatsIdentificacio.substring(27, 27 +
digitsPercentatge));
                if (27+digitsPercentatge+3 < resultatsIdentificacio.length())
                    resultatsIdentificacio =
resultatsIdentificacio.substring(27+digitsPercentatge+3,
resultatsIdentificacio.length());
                else
                    resultatsIdentificacio = "";
            }
            else if (resultatsIdentificacio.length() > 11 && resultatsIdentificacio.substring(0,
11).compareTo("Cicada orni") == 0) {
                while(resultatsIdentificacio.charAt(13+digitsPercentatge) != '%')
digitsPercentatge++;
                llista.add((resultatsIdentificacio.charAt(11) != '*' ? "Cicada orni" : "Cicada
orni*") + " " + resultatsIdentificacio.substring(13, 13+digitsPercentatge));
                if (13+digitsPercentatge+3 < resultatsIdentificacio.length())
                    resultatsIdentificacio =
resultatsIdentificacio.substring(13+digitsPercentatge+3,
resultatsIdentificacio.length());
                else
                    resultatsIdentificacio = "";
            }
            else if (resultatsIdentificacio.length() > 14 && resultatsIdentificacio.substring(0,
14).compareTo("Cicadatra atra") == 0) {
                while(resultatsIdentificacio.charAt(16+digitsPercentatge) != '%')
digitsPercentatge++;
                llista.add((resultatsIdentificacio.charAt(14) != '*' ? "Cicadatra atra" :
"Cicadatra atra*") + " " + resultatsIdentificacio.substring(16, 16 + digitsPercentatge));
                if (16+digitsPercentatge+3 < resultatsIdentificacio.length())
                    resultatsIdentificacio =
resultatsIdentificacio.substring(16+digitsPercentatge+3,
resultatsIdentificacio.length());
                else
                    resultatsIdentificacio = "";
            }
            else if (resultatsIdentificacio.length() > 21 && resultatsIdentificacio.substring(0,
21).compareTo("Cicadetta brevipennis") == 0) {
                while(resultatsIdentificacio.charAt(23+digitsPercentatge) != '%')
digitsPercentatge++;
                llista.add((resultatsIdentificacio.charAt(21) != '*' ? "Cicadetta brevipennis" :
"Cicadetta brevipennis*") + " " + resultatsIdentificacio.substring(23, 23 +
digitsPercentatge));
                if (23+digitsPercentatge+3 < resultatsIdentificacio.length())
                    resultatsIdentificacio =
resultatsIdentificacio.substring(23+digitsPercentatge+3,
resultatsIdentificacio.length());
                else
                    resultatsIdentificacio = "";
```

```
        }
        else if (resultatsIdentificacio.length() > 22 && resultatsIdentificacio.substring(0,
22).compareTo("Cicadetta cerdaniensis") == 0) {
            while(resultatsIdentificacio.charAt(24+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(22) != '*' ? "Cicadetta cerdaniensis" :
"Cicadetta cerdaniensis*") + " " + resultatsIdentificacio.substring(24, 24 +
digitsPercentatge));
            if (24+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(24+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 17 && resultatsIdentificacio.substring(0,
17).compareTo("Hilaphura varipes") == 0) {
            while(resultatsIdentificacio.charAt(19+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(17) != '*' ? "Hilaphura varipes" :
"Hilaphura varipes*") + " " + resultatsIdentificacio.substring(19, 19 +
digitsPercentatge));
            if (19+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(19+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 17 && resultatsIdentificacio.substring(0,
17).compareTo("Lyristes plebejus") == 0) {
            while(resultatsIdentificacio.charAt(19+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(17) != '*' ? "Lyristes plebejus" :
"Lyristes plebejus*") + " " + resultatsIdentificacio.substring(19, 19 +
digitsPercentatge));
            if (19+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(19+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 23 && resultatsIdentificacio.substring(0,
23).compareTo("Tettigettalna argentata") == 0) {
            while(resultatsIdentificacio.charAt(25+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(23) != '*' ? "Tettigettalna argentata" :
"Tettigettalna argentata*") + " " + resultatsIdentificacio.substring(25, 25 +
digitsPercentatge));
            if (25+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(25+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 19 && resultatsIdentificacio.substring(0,
19).compareTo("Tettigettula pygmea") == 0) {
            while(resultatsIdentificacio.charAt(21+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(19) != '*' ? "Tettigettula pygmea" :
"Tettigettula pygmea*") + " " + resultatsIdentificacio.substring(21, 21 +
digitsPercentatge));
            if (21+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(21+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 27 && resultatsIdentificacio.substring(0,
27).compareTo("Tibicina corsica fairmairei") == 0) {
            while(resultatsIdentificacio.charAt(29+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(27) != '*' ? "Tibicina corsica fairmairei"
: "Tibicina corsica fairmairei*") + " " + resultatsIdentificacio.substring(29, 29 +
digitsPercentatge));
            if (29+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(29+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 18 && resultatsIdentificacio.substring(0,
18).compareTo("Tibicina garricola") == 0) {
            while(resultatsIdentificacio.charAt(21+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(18) != '*' ? "Tibicina garricola" :
"Tibicina garricola*") + " " + resultatsIdentificacio.substring(20, 20 +
digitsPercentatge));
            if (20+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(20+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 19 && resultatsIdentificacio.substring(0,
19).compareTo("Tibicina haematodes") == 0) {
            while(resultatsIdentificacio.charAt(21+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(19) != '*' ? "Tibicina haematodes" :
"Tibicina haematodes*") + " " + resultatsIdentificacio.substring(21, 21 +
digitsPercentatge));
            if (21+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(21+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 22 && resultatsIdentificacio.substring(0,
22).compareTo("Tibicina quadrisignata") == 0) {
            while(resultatsIdentificacio.charAt(24+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(22) != '*' ? "Tibicina quadrisignata" :
"Tibicina quadrisignata*") + " " + resultatsIdentificacio.substring(24, 24 +
digitsPercentatge));
            if (24+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(24+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else if (resultatsIdentificacio.length() > 18 && resultatsIdentificacio.substring(0,
18).compareTo("Tibicina tomentosa") == 0) {
            while(resultatsIdentificacio.charAt(20+digitsPercentatge)        !=        '%')
digitsPercentatge++;
            llista.add((resultatsIdentificacio.charAt(18) != '*' ? "Tibicina tomentosa" :
"Tibicina tomentosa*") + " " + resultatsIdentificacio.substring(20, 20 +
digitsPercentatge));
            if (20+digitsPercentatge+3 < resultatsIdentificacio.length())
              resultatsIdentificacio                                                =
resultatsIdentificacio.substring(20+digitsPercentatge+3,
resultatsIdentificacio.length());
            else
              resultatsIdentificacio = "";
        }
        else continua = false;
      }
      return llista;
    }

    private void botoReprodueixClic() {
      botoReprodueix.setVisibility(View.GONE);
      botoAnalitza.setVisibility(View.GONE);
      botoDescarta.setVisibility(View.GONE);
      botoAtura.setVisibility(View.VISIBLE);
      microfonFonsBlanc.setVisibility(View.GONE);
      rellotgeDeSorra.setVisibility(View.GONE);
      reproduccio.setVisibility(View.VISIBLE);
      carregant.setVisibility(View.VISIBLE);
      carregant.setAnimation(rotacio);
      reproduccioEnCurs.setVisibility(View.VISIBLE);
      reproduccioEnCurs.setAnimation(parpalleig);

      iniciaReproduccio();
    }

    private void botoAturaClic() {
      botoReprodueix.setVisibility(View.VISIBLE);
      botoAnalitza.setVisibility(View.VISIBLE);
      botoDescarta.setVisibility(View.VISIBLE);
      botoAtura.setVisibility(View.GONE);
      microfonFonsBlanc.setVisibility(View.VISIBLE);
      rellotgeDeSorra.setVisibility(View.GONE);
      reproduccio.setVisibility(View.GONE);
      carregant.clearAnimation();
      carregant.setVisibility(View.GONE);
      reproduccioEnCurs.setVisibility(View.GONE);
      reproduccioEnCurs.clearAnimation();

      acabaReproduccio();
    }

    private void iniciaReproduccio() {
      reproductor = new MediaPlayer();
```

```
        try {
            reproductor.setDataSource(nomFitxer);
            reproductor.prepare();
            reproductor.start();
            reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener()
            {
                @Override
                public void onCompletion(MediaPlayer mp)
                {
                    botoAturaClic();
                }
            });
        } catch (IOException e) {
            Log.e("Error", "El mètode prepare() ha fallat.");
        }
    }

    private void acabaReproduccio() {
        reproductor.release();
        reproductor = null;
    }

    private void iniciaEnregistrament() {
        enregistrador = new MediaRecorder();
        enregistrador.setAudioSource(MediaRecorder.AudioSource.MIC);
        enregistrador.setOutputFormat(MediaRecorder.OutputFormat.MPEG_4);
        enregistrador.setOutputFile(nomFitxer);
        enregistrador.setAudioEncoder(MediaRecorder.AudioEncoder.AMR_NB);

        try {
            enregistrador.prepare();
        } catch (IOException e) {
            Log.e("Error", "El mètode prepare() ha fallat.");
        }

        enregistrador.start();
    }

    private void acabaEnregistrament() {
        enregistrador.stop();
        enregistrador.release();
        enregistrador = null;
    }

    public EnregistradorAudio() {
        nomFitxer = Environment.getExternalStorageDirectory().getAbsolutePath();
        nomFitxer += "/audiorecordtest.mp4";
    }

    @Override
    public void onPause() {
        super.onPause();
        if (enregistrador != null) {
            enregistrador.release();
            enregistrador = null;
        }

        if (reproductor != null) {
            reproductor.release();
            reproductor = null;
        }
    }

    private void microfonFonsBlancClic() {
        microfonFonsBlanc.setVisibility(View.GONE);
        rellotgeDeSorra.setVisibility(View.GONE);
        microfon.setVisibility(View.VISIBLE);
        carregant.setVisibility(View.VISIBLE);
        carregant.startAnimation(rotacio);
        gravacioEnCurs.setVisibility(View.VISIBLE);
        gravacioEnCurs.setAnimation(parpalleig);

        enregistradorWav.iniciaEnregistrament();
    }

    private void microfonClic() {
        microfonFonsBlanc.setVisibility(View.VISIBLE);
        microfon.setVisibility(View.GONE);
        rellotgeDeSorra.setVisibility(View.GONE);
        carregant.clearAnimation();
        gravacioEnCurs.clearAnimation();
        carregant.setVisibility(View.GONE);
        gravacioEnCurs.setVisibility(View.GONE);

        microfonFonsBlanc.setEnabled(false);
        botoReprodueix.setVisibility(View.VISIBLE);
        botoAnalitza.setVisibility(View.VISIBLE);
        botoDescarta.setVisibility(View.VISIBLE);

        nomFitxer = enregistradorWav.acabaEnregistrament();
    }

    @Override
    public void onClick(View view) {
    }
}
```

EnregistradorWav.java:

```
package cigales.cicadaapp;


import android.media.AudioRecord;
import android.media.MediaRecorder;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;
import java.text.SimpleDateFormat;
import java.util.Date;

import android.media.AudioFormat;
import android.os.Environment;
import android.util.Log;

/**
 EnregistradorWav és la classe encarregada d'enregistrar i desar un fitxer .wav
mitjançant l'enregistrador del mòbil.
 */

public class EnregistradorWav {
    private static final int RECORDER_BPP = 16;
    private static final String AUDIO_RECORDER_FILE_EXT_WAV = ".wav";
    private static final String AUDIO_RECORDER_FOLDER = "AudioRecorder";
    private static final String AUDIO_RECORDER_TEMP_FILE = "record_temp.raw";
    private static final int RECORDER_SAMPLERATE = 44100;
    private static final int RECORDER_CHANNELS = AudioFormat.CHANNEL_IN_MONO;
    private static final int RECORDER_AUDIO_ENCODING =
AudioFormat.ENCODING_PCM_16BIT;

    private AudioRecord enregistrador = null;
    private int midaBuffer = AudioRecord.getMinBufferSize(RECORDER_SAMPLERATE,
            2, RECORDER_AUDIO_ENCODING); //2 -> Mono, 3 -> Stereo
    private Thread filEnregistrador = null;
    private boolean estaGravant = false;


    public void iniciaEnregistrament(){
        enregistrador = new AudioRecord(MediaRecorder.AudioSource.MIC,
                RECORDER_SAMPLERATE,
RECORDER_CHANNELS,RECORDER_AUDIO_ENCODING, midaBuffer);

        int estat = enregistrador.getState();

        if(estat==1) {
            enregistrador.startRecording();
            estaGravant = true;

            filEnregistrador = new Thread(new Runnable() {
                @Override
                public void run() {
                    escriuDadesAudioAFitxer();
                }
            }, "AudioRecorder Thread");

            filEnregistrador.start();
        }
    }

    private void escriuDadesAudioAFitxer(){
        byte dades[] = new byte[midaBuffer];
        String nomFitxer = obtinguesNomFitxerTemporal();
        FileOutputStream fluxSortida = null;

        try {
            fluxSortida = new FileOutputStream(nomFitxer);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
            Log.e("Error", e.getMessage().toString());
        }

        int llegits = 0;

        if(null != fluxSortida){
```

```
        while(estaGravant){
            llegits = enregistrador.read(dades, 0, midaBuffer);

            if(AudioRecord.ERROR_INVALID_OPERATION != llegits){
                try {
                    fluxSortida.write(dades);
                } catch (IOException e) {
                    e.printStackTrace();
                    Log.e("Error", e.getMessage().toString());
                }
            }
        }

        try {
            fluxSortida.close();
        } catch (IOException e) {
            e.printStackTrace();
            Log.e("Error", e.getMessage().toString());
        }
    }
}

private String obtinguesNomFitxerTemporal(){
    String rutaFitxer = Environment.getExternalStorageDirectory().getPath();
    File fitxer = new File(rutaFitxer,AUDIO_RECORDER_FOLDER);

    if(!fitxer.exists()){
        fitxer.mkdirs();
    }

    File fitxerTemporal = new File(rutaFitxer,AUDIO_RECORDER_TEMP_FILE);

    if(fitxerTemporal.exists())
        fitxerTemporal.delete();

    return (fitxer.getAbsolutePath() + "/" + AUDIO_RECORDER_TEMP_FILE);
}

public String acabaEnregistrament() {
    if(null != enregistrador){
        estaGravant = false;

        int i = enregistrador.getState();
        if(i==1)
            enregistrador.stop();
        enregistrador.release();

        enregistrador = null;
        filEnregistrador = null;
    }

    String nomFitxer = obtinguesNomFitxer();

    copiaFitxerWav(obtinguesNomFitxerTemporal(), nomFitxer);
    eliminaFitxerTemporal();

    BufferedReader lector = null;
    try {
        lector = new BufferedReader(new FileReader(nomFitxer));
    } catch (FileNotFoundException e) {
        e.printStackTrace();
        Log.e("Error", e.getMessage().toString());
    }
    File mFitxer = new File(nomFitxer);
    int llargadaFitxer = (int) mFitxer.length();
    byte[] buffer = new byte[llargadaFitxer];
    try {
        int valor=0;
        int posicio = 0;

        //Llegeix fins al final del flux.
        while((valor = lector.read()) != -1)
        {
            //Converteix un enter en un caràcter.
            byte c = (byte)valor;
            buffer[posicio] = c;
            posicio++;
        }
        lector.close();
    } catch (IOException e) {
        e.printStackTrace();
        Log.e("Error", e.getMessage().toString());
    }

    return nomFitxer;
}

private void eliminaFitxerTemporal() {
    File fitxer = new File(obtinguesNomFitxerTemporal());

    fitxer.delete();
}

private void copiaFitxerWav(String nomFitxerEntrada, String nomFitxerSortida){
    FileInputStream entrada = null;
    FileOutputStream sortida = null;
    long llargadaTotalAudio = 0;
    long llargadaTotalDades = llargadaTotalAudio + 36;
    long frequenciaMostreig = RECORDER_SAMPLERATE;
    int canals = 1;
    long ratioBytes = RECORDER_BPP * RECORDER_SAMPLERATE * canals/8;

    byte[] dades = new byte[midaBuffer];

    try {
        entrada = new FileInputStream(nomFitxerEntrada);
        sortida = new FileOutputStream(nomFitxerSortida);
        llargadaTotalAudio = entrada.getChannel().size();
        llargadaTotalDades = llargadaTotalAudio + 36;

        escriuCapcaleraFitxerWav(sortida, llargadaTotalAudio, llargadaTotalDades,
                frequenciaMostreig, canals, ratioBytes);

        while(entrada.read(dades) != -1){
            sortida.write(dades);
        }

        entrada.close();
        sortida.close();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
        Log.e("Error", e.getMessage().toString());
    } catch (IOException e) {
        e.printStackTrace();
        Log.e("Error", e.getMessage().toString());
    }
}

private String obtinguesNomFitxer(){
    String rutaFitxer = Environment.getExternalStorageDirectory().getPath().replace("/AudioRecorder", "");
    File fitxer = new File(rutaFitxer,"CicadaAppRecordings");

    if(!fitxer.exists()){
        fitxer.mkdirs();
    }

    SimpleDateFormat formatData = new SimpleDateFormat("ddMMyyyy_HHmmss");
    String dataIHoraActual = formatData.format(new Date());

    return (fitxer.getAbsolutePath() + "/" + dataIHoraActual + AUDIO_RECORDER_FILE_EXT_WAV);
}

private void escriuCapcaleraFitxerWav(
        FileOutputStream sortida, long llargadaTotalAudio,
        long llargadaTotalDades, long frequenciaMostreig, int canals,
        long ratioBytes) throws IOException {

    byte[] capcalera = new byte[44];

    capcalera[0] = 'R'; //Capçalera RIFF/WAVE
    capcalera[1] = 'I';
    capcalera[2] = 'F';
    capcalera[3] = 'F';
    capcalera[4] = (byte) (llargadaTotalDades & 0xff);
    capcalera[5] = (byte) ((llargadaTotalDades >> 8) & 0xff);
    capcalera[6] = (byte) ((llargadaTotalDades >> 16) & 0xff);
    capcalera[7] = (byte) ((llargadaTotalDades >> 24) & 0xff);
    capcalera[8] = 'W';
    capcalera[9] = 'A';
    capcalera[10] = 'V';
    capcalera[11] = 'E';
    capcalera[12] = 'f'; //Fragment 'fmt '
    capcalera[13] = 'm';
    capcalera[14] = 't';
    capcalera[15] = ' ';
    capcalera[16] = 16; //4 bytes: mida del fragment 'fmt '
    capcalera[17] = 0;
    capcalera[18] = 0;
    capcalera[19] = 0;
    capcalera[20] = 1; //Format = 1
    capcalera[21] = 0;
    capcalera[22] = (byte) canals;
    capcalera[23] = 0;
    capcalera[24] = (byte) (frequenciaMostreig & 0xff);
    capcalera[25] = (byte) ((frequenciaMostreig >> 8) & 0xff);
    capcalera[26] = (byte) ((frequenciaMostreig >> 16) & 0xff);
    capcalera[27] = (byte) ((frequenciaMostreig >> 24) & 0xff);
    capcalera[28] = (byte) (ratioBytes & 0xff);
    capcalera[29] = (byte) ((ratioBytes >> 8) & 0xff);
```

```
        capcalera[30] = (byte) ((ratioBytes >> 16) & 0xff);
        capcalera[31] = (byte) ((ratioBytes >> 24) & 0xff);
        capcalera[32] = (byte) (2 * 16 / 8); // block align
        capcalera[33] = 0;
        capcalera[34] = RECORDER_BPP; //Bits per mostra
        capcalera[35] = 0;
        capcalera[36] = 'd';
        capcalera[37] = 'a';
        capcalera[38] = 't';
        capcalera[39] = 'a';
        capcalera[40] = (byte) (llargadaTotalAudio & 0xff);
        capcalera[41] = (byte) ((llargadaTotalAudio >> 8) & 0xff);
        capcalera[42] = (byte) ((llargadaTotalAudio >> 16) & 0xff);
        capcalera[43] = (byte) ((llargadaTotalAudio >> 24) & 0xff);

        sortida.write(capcalera, 0, 44);
    }

    public EnregistradorWav() {
    }
}
```

ExploradorFitxers.java:

```
package cigales.cicadaapp;

import java.io.File;
import java.sql.Date;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.text.DateFormat;

import android.content.Context;
import android.os.Bundle;
import android.app.ListActivity;
import android.content.Intent;
import android.util.Log;
import android.view.View;
import android.widget.ListView;
import android.widget.Toast;

/**
 ExploradorFitxers és la classe que gestiona la pantalla de l'explorador de fitxers que s'utilitza a l'hora d'analitzar fitxers enregistrats amb antelació.
 */

public class ExploradorFitxers extends ListActivity {

    private File directoriActual;
    private AdaptadorListViewFitxers adaptador;

    Context context = this;

    Configuracio configuracio = new Configuracio(context);

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        //Llegeix la carpeta que s'ha d'obrir, que serà l'última des d'on s'hagi analitzat un fitxer.
        String[] parametresLlegits = configuracio.llegeix(3, "configuració.txt");

        try {
            if (parametresLlegits.length > 1 && new File(parametresLlegits[1]).isDirectory())
                directoriActual = new File(parametresLlegits[1]);
            else directoriActual = new File("/sdcard/");
        }
        catch (Exception e) {
            directoriActual = new File("/sdcard/");
            Log.e("Error", "Excepció en obrir l'última carpeta des d'on s'ha analitzat un fitxer");
        }

        emplena(directoriActual);
    }

    //Llegeix i mostra tots els fitxers i carpetes que es trobin al directori actual.
    private void emplena(File f)
    {
        File[]llistaFitxers = f.listFiles();
        this.setTitle(R.string.DirectoriActual + ": " + f.getName());
        List<Fitxer>carpeta = new ArrayList<Fitxer>();
        List<Fitxer>fitxers = new ArrayList<Fitxer>();
        try{
            for(File ff: llistaFitxers)
            {
                Date ultimaDataModificacio = new Date(ff.lastModified());
                DateFormat formatejador = DateFormat.getDateTimeInstance();
                String dataModificacio = formatejador.format(ultimaDataModificacio);
                if(ff.isDirectory()){

                    File[] bufferFitxers = ff.listFiles();
                    int buffer = 0;
                    if(bufferFitxers != null){
                        buffer = bufferFitxers.length;
                    }
                    else buffer = 0;
                    String nFitxers = String.valueOf(buffer);
                    if(buffer == 0) nFitxers = nFitxers + " " + R.string.Fitxer;
                    else nFitxers = nFitxers + " " + R.string.Fitxers;

                    carpeta.add(new Fitxer(ff.getName(), nFitxers, dataModificacio, ff.getAbsolutePath(), "carpeta"));
                }
                else if (ff.getName().substring(ff.getName().length()-4, ff.getName().length()).compareTo(".wav") == 0)
                {
                    fitxers.add(new Fitxer(ff.getName(), ff.length() + " bytes", dataModificacio, ff.getAbsolutePath(), "fitxer_valid"));
                }
                else
                {
                    fitxers.add(new Fitxer(ff.getName(), ff.length() + " bytes", dataModificacio, ff.getAbsolutePath(), "fitxer"));
                }
            }
        }catch(Exception e)
        {
            Log.e("Error", e.getMessage().toString());
        }
        Collections.sort(carpeta);
        Collections.sort(fitxers);
        carpeta.addAll(fitxers);

        if(!f.getName().equalsIgnoreCase("sdcard")) {
            carpeta.add(0, new Fitxer("...", f.getParentFile().getName(), "", f.getParent(), "carpeta_contenidora"));
        }

        adaptador = new AdaptadorListViewFitxers(ExploradorFitxers.this,R.layout.llista_fitxers,carpeta);
        this.setListAdapter(adaptador);
    }

    //Si es clica sobre la icona de la carpeta contenidora o sobre alguna carpeta continguda al directori actual, s'obre aquesta, si es clica un fitxer .wav s'intenta analitzar-lo i si es clica un fitxer d'un altre format s'avisa que el fitxer no és vàlid per ser analitzat.
    @Override
    protected void onListItemClick(ListView l, View v, int posicio, long id) {
        super.onListItemClick(l, v, posicio, id);
        Fitxer f = adaptador.getItem(posicio);

if(f.getImatge().equalsIgnoreCase("carpeta")||f.getImatge().equalsIgnoreCase("carpeta_contenidora")){
            directoriActual = new File(f.getRuta());
            emplena(directoriActual);
        }
        else if (f.getImatge().equalsIgnoreCase("fitxer_valid"))
        {
            Intent intent = new Intent("cigales.cicadaapp.EnregistradorAudio");
            Bundle b = new Bundle();
            b.putString("fitxerPerAnalitzar", f.getRuta());
            intent.putExtras(b);

            String[] informacions = configuracio.llegeix(3, "configuració.txt");
            if (informacions.length > 1) informacions[1] = f.getRuta().substring(0, f.getRuta().length() - f.getNom().length());
            configuracio.escriu(informacions, "configuració.txt");

            finish();
            startActivity(intent);
        }
        else
        {
            onFileClick(f);
        }
    }
    private void onFileClick(Fitxer o)
    {
        Toast.makeText(this, context.getResources().getString(R.string.FitxerNoValid) + ": " + o.getNom() + "; " + context.getResources().getString(R.string.SHaDeSeleccionarUnFitxerWav) + ".", Toast.LENGTH_LONG).show();
    }
}
```

FitxaEspecie.java:

```
package cigales.cicadaapp;

import android.content.Context;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.ImageView;

/**
 FitxaEspecie és la classe que gestiona, de manera genèrica, la pantalla que mostra la
informació base de cada una de les espècies de cigala tractades a
 l'aplicació.
 */


public class FitxaEspecie extends AppCompatActivity implements View.OnClickListener
{

   String especie = "";

   ImageView cicOrnPlay1;
   ImageView cicOrnPlay2;
   ImageView cicBarPlay1;
   ImageView cicBarPlay2;
   ImageView cicAtrPlay1;
   ImageView cicAtrPlay2;
   ImageView cicAtrPlay3;
   ImageView cicBrePlay1;
   ImageView cicBrePlay2;
   ImageView cicBrePlay3;
   ImageView cicCerPlay1;
   ImageView hilVarPlay1;
   ImageView lyrPlePlay1;
   ImageView lyrPlePlay2;
   ImageView tetArgPlay1;
   ImageView tetPygPlay1;
   ImageView tetPygPlay2;
   ImageView tibCorPlay1;
   ImageView tibCorPlay2;
   ImageView tibCorPlay3;
   ImageView tibGarPlay1;
   ImageView tibHaePlay1;
   ImageView tibQuaPlay1;
   ImageView tibQuaPlay2;
   ImageView tibQuaPlay3;
   ImageView tibTomPlay1;
   ImageView tibTomPlay2;
   ImageView tibTomPlay3;


   ImageView cicOrnPausa1;
   ImageView cicOrnPausa2;
   ImageView cicBarPausa1;
   ImageView cicBarPausa2;
   ImageView cicAtrPausa1;
   ImageView cicAtrPausa2;
   ImageView cicAtrPausa3;
   ImageView cicBrePausa1;
   ImageView cicBrePausa2;
   ImageView cicBrePausa3;
   ImageView cicCerPausa1;
   ImageView hilVarPausa1;
   ImageView lyrPlePausa1;
   ImageView lyrPlePausa2;
   ImageView tetArgPausa1;
   ImageView tetPygPausa1;
   ImageView tetPygPausa2;
   ImageView tibCorPausa1;
   ImageView tibCorPausa2;
   ImageView tibCorPausa3;
   ImageView tibGarPausa1;
   ImageView tibHaePausa1;
   ImageView tibQuaPausa1;
   ImageView tibQuaPausa2;
   ImageView tibQuaPausa3;
   ImageView tibTomPausa1;
   ImageView tibTomPausa2;
   ImageView tibTomPausa3;

   int posicio1;
   int posicio2;
   int posicio3;

   MediaPlayer reproductor;

   Context context = this;

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);

      posicio1 = 0;
      posicio2 = 0;
      posicio3 = 0;

      Bundle b = getIntent().getExtras();
      if(b != null) {
         especie = b.getString("especie");
         obreFitxaEspecie(especie);

         if (especie.compareTo("Cicada barbara lusitanica") == 0) {
            cicBarPlay1 = (ImageView)findViewById(R.id.CicBar_Cant1_Play);
            cicBarPlay2 = (ImageView)findViewById(R.id.CicBar_Cant2_Play);
            cicBarPausa1 = (ImageView)findViewById(R.id.CicBar_Cant1_Pausa);
            cicBarPausa2 = (ImageView)findViewById(R.id.CicBar_Cant2_Pausa);

            cicBarPlay1.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View v) {
                  reproductor = MediaPlayer.create(context,
R.raw.cicada_barbara_lusitanica_wav1);
                  reproductor.seekTo(posicio1);
                  posicio1 = 0;
                  reproductor.start();
                  reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                     @Override
                     public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        cicBarPlay1.setVisibility(View.VISIBLE);
                        cicBarPausa1.setVisibility(View.GONE);
                     }
                  });
                  cicBarPlay1.setVisibility(View.GONE);
                  cicBarPausa1.setVisibility(View.VISIBLE);
               }
            });

            cicBarPlay2.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View v) {
                  reproductor = MediaPlayer.create(context,
R.raw.cicada_barbara_lusitanica_wav2);
                  reproductor.seekTo(posicio2);
                  posicio2 = 0;
                  reproductor.start();
                  reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                     @Override
                     public void onCompletion(MediaPlayer mp) {
                        posicio2 = 0;
                        cicBarPlay2.setVisibility(View.VISIBLE);
                        cicBarPausa2.setVisibility(View.GONE);
                     }
                  });
                  cicBarPlay2.setVisibility(View.GONE);
                  cicBarPausa2.setVisibility(View.VISIBLE);
               }
            });

            cicBarPausa1.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View v) {
                  reproductor.pause();
                  posicio1 = reproductor.getCurrentPosition();
                  cicBarPlay1.setVisibility(View.VISIBLE);
                  cicBarPausa1.setVisibility(View.GONE);
               }
            });

            cicBarPausa2.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View v) {
                  reproductor.pause();
                  posicio2 = reproductor.getCurrentPosition();
                  cicBarPlay2.setVisibility(View.VISIBLE);
                  cicBarPausa2.setVisibility(View.GONE);
               }
            });
         }
         else if (especie.compareTo("Cicada orni") == 0) {
            cicOrnPlay1 = (ImageView)findViewById(R.id.CicOrn_Cant1_Play);
            cicOrnPlay2 = (ImageView)findViewById(R.id.CicOrn_Cant2_Play);
            cicOrnPausa1 = (ImageView)findViewById(R.id.CicOrn_Cant1_Pausa);
            cicOrnPausa2 = (ImageView)findViewById(R.id.CicOrn_Cant2_Pausa);
```

```
        cicOrnPlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context, R.raw.cicada_orni_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        cicOrnPlay1.setVisibility(View.VISIBLE);
                        cicOrnPausa1.setVisibility(View.GONE);
                    }
                });
                cicOrnPlay1.setVisibility(View.GONE);
                cicOrnPausa1.setVisibility(View.VISIBLE);
            }
        });

        cicOrnPlay2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context, R.raw.cicada_orni_wav2);
                reproductor.seekTo(posicio2);
                posicio2 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio2 = 0;
                        cicOrnPlay2.setVisibility(View.VISIBLE);
                        cicOrnPausa2.setVisibility(View.GONE);
                    }
                });
                cicOrnPlay2.setVisibility(View.GONE);
                cicOrnPausa2.setVisibility(View.VISIBLE);
            }
        });

        cicOrnPausa1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                cicOrnPlay1.setVisibility(View.VISIBLE);
                cicOrnPausa1.setVisibility(View.GONE);
            }
        });

        cicOrnPausa2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio2 = reproductor.getCurrentPosition();
                cicOrnPlay2.setVisibility(View.VISIBLE);
                cicOrnPausa2.setVisibility(View.GONE);
            }
        });
    }
    else if (especie.compareTo("Cicadatra atra") == 0) {
        cicAtrPlay1 = (ImageView)findViewById(R.id.CicAtr_Cant1_Play);
        cicAtrPlay2 = (ImageView)findViewById(R.id.CicAtr_Cant2_Play);
        cicAtrPlay3 = (ImageView)findViewById(R.id.CicAtr_Cant3_Play);
        cicAtrPausa1 = (ImageView)findViewById(R.id.CicAtr_Cant1_Pausa);
        cicAtrPausa2 = (ImageView)findViewById(R.id.CicAtr_Cant2_Pausa);
        cicAtrPausa3 = (ImageView)findViewById(R.id.CicAtr_Cant3_Pausa);

        cicAtrPlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context, R.raw.cicadatra_atra_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        cicAtrPlay1.setVisibility(View.VISIBLE);
                        cicAtrPausa1.setVisibility(View.GONE);
                    }
                });
                cicAtrPlay1.setVisibility(View.GONE);
                cicAtrPausa1.setVisibility(View.VISIBLE);
            }
        });

        cicAtrPlay2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context, R.raw.cicadatra_atra_wav2);
                reproductor.seekTo(posicio2);
                posicio2 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio2 = 0;
                        cicAtrPlay2.setVisibility(View.VISIBLE);
                        cicAtrPausa2.setVisibility(View.GONE);
                    }
                });
                cicAtrPlay2.setVisibility(View.GONE);
                cicAtrPausa2.setVisibility(View.VISIBLE);
            }
        });

        cicAtrPlay3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context, R.raw.cicadatra_atra_wav3);
                reproductor.seekTo(posicio3);
                posicio3 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio3 = 0;
                        cicAtrPlay3.setVisibility(View.VISIBLE);
                        cicAtrPausa3.setVisibility(View.GONE);
                    }
                });
                cicAtrPlay3.setVisibility(View.GONE);
                cicAtrPausa3.setVisibility(View.VISIBLE);
            }
        });

        cicAtrPausa1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                cicAtrPlay1.setVisibility(View.VISIBLE);
                cicAtrPausa1.setVisibility(View.GONE);
            }
        });

        cicAtrPausa2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio2 = reproductor.getCurrentPosition();
                cicAtrPlay2.setVisibility(View.VISIBLE);
                cicAtrPausa2.setVisibility(View.GONE);
            }
        });

        cicAtrPausa3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio3 = reproductor.getCurrentPosition();
                cicAtrPlay3.setVisibility(View.VISIBLE);
                cicAtrPausa3.setVisibility(View.GONE);
            }
        });
    }
    else if (especie.compareTo("Cicadetta brevipennis") == 0) {
        cicBrePlay1 = (ImageView)findViewById(R.id.CicBre_Cant1_Play);
        cicBrePlay2 = (ImageView)findViewById(R.id.CicBre_Cant2_Play);
        cicBrePlay3 = (ImageView)findViewById(R.id.CicBre_Cant3_Play);
        cicBrePausa1 = (ImageView)findViewById(R.id.CicBre_Cant1_Pausa);
        cicBrePausa2 = (ImageView)findViewById(R.id.CicBre_Cant2_Pausa);
        cicBrePausa3 = (ImageView)findViewById(R.id.CicBre_Cant3_Pausa);

        cicBrePlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.cicadetta_brevipennis_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
```

```
                public void onCompletion(MediaPlayer mp) {
                    posicio1 = 0;
                    cicBrePlay1.setVisibility(View.VISIBLE);
                    cicBrePausa1.setVisibility(View.GONE);
                }
            });
            cicBrePlay1.setVisibility(View.GONE);
            cicBrePausa1.setVisibility(View.VISIBLE);
        }
    });

    cicBrePlay2.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor = MediaPlayer.create(context,
R.raw.cicadetta_brevipennis_wav2);
            reproductor.seekTo(posicio2);
            posicio2 = 0;
            reproductor.start();
            reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                @Override
                public void onCompletion(MediaPlayer mp) {
                    posicio2 = 0;
                    cicBrePlay2.setVisibility(View.VISIBLE);
                    cicBrePausa2.setVisibility(View.GONE);
                }
            });
            cicBrePlay2.setVisibility(View.GONE);
            cicBrePausa2.setVisibility(View.VISIBLE);
        }
    });

    cicBrePlay3.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor = MediaPlayer.create(context,
R.raw.cicadetta_brevipennis_wav3);
            reproductor.seekTo(posicio3);
            posicio3 = 0;
            reproductor.start();
            reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                @Override
                public void onCompletion(MediaPlayer mp) {
                    posicio3 = 0;
                    cicBrePlay3.setVisibility(View.VISIBLE);
                    cicBrePausa3.setVisibility(View.GONE);
                }
            });
            cicBrePlay3.setVisibility(View.GONE);
            cicBrePausa3.setVisibility(View.VISIBLE);
        }
    });

    cicBrePausa1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor.pause();
            posicio1 = reproductor.getCurrentPosition();
            cicBrePlay1.setVisibility(View.VISIBLE);
            cicBrePausa1.setVisibility(View.GONE);
        }
    });

    cicBrePausa2.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor.pause();
            posicio2 = reproductor.getCurrentPosition();
            cicBrePlay2.setVisibility(View.VISIBLE);
            cicBrePausa2.setVisibility(View.GONE);
        }
    });

    cicBrePausa3.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor.pause();
            posicio3 = reproductor.getCurrentPosition();
            cicBrePlay3.setVisibility(View.VISIBLE);
            cicBrePausa3.setVisibility(View.GONE);
        }
    });
}
else if (especie.compareTo("Cicadetta cerdaniensis") == 0) {
    cicCerPlay1 = (ImageView)findViewById(R.id.CicCer_Cant1_Play);
    cicCerPausa1 = (ImageView)findViewById(R.id.CicCer_Cant1_Pausa);

    cicCerPlay1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor = MediaPlayer.create(context,
R.raw.cicadetta_cerdaniensis_wav1);
            reproductor.seekTo(posicio1);
            posicio1 = 0;
            reproductor.start();
            reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                @Override
                public void onCompletion(MediaPlayer mp) {
                    posicio1 = 0;
                    cicCerPlay1.setVisibility(View.VISIBLE);
                    cicCerPausa1.setVisibility(View.GONE);
                }
            });
            cicCerPlay1.setVisibility(View.GONE);
            cicCerPausa1.setVisibility(View.VISIBLE);
        }
    });

    cicCerPausa1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor.pause();
            posicio1 = reproductor.getCurrentPosition();
            cicCerPlay1.setVisibility(View.VISIBLE);
            cicCerPausa1.setVisibility(View.GONE);
        }
    });
}
else if (especie.compareTo("Hilaphura varipes") == 0) {
    hilVarPlay1 = (ImageView)findViewById(R.id.HilVar_Cant1_Play);
    hilVarPausa1 = (ImageView)findViewById(R.id.HilVar_Cant1_Pausa);

    hilVarPlay1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor = MediaPlayer.create(context,
R.raw.hilaphura_varipes_wav1);
            reproductor.seekTo(posicio1);
            posicio1 = 0;
            reproductor.start();
            reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                @Override
                public void onCompletion(MediaPlayer mp) {
                    posicio1 = 0;
                    hilVarPlay1.setVisibility(View.VISIBLE);
                    hilVarPausa1.setVisibility(View.GONE);
                }
            });
            hilVarPlay1.setVisibility(View.GONE);
            hilVarPausa1.setVisibility(View.VISIBLE);
        }
    });

    hilVarPausa1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor.pause();
            posicio1 = reproductor.getCurrentPosition();
            hilVarPlay1.setVisibility(View.VISIBLE);
            hilVarPausa1.setVisibility(View.GONE);
        }
    });
}
else if (especie.compareTo("Lyristes plebejus") == 0) {
    lyrPlePlay1 = (ImageView)findViewById(R.id.LyrPle_Cant1_Play);
    lyrPlePlay2 = (ImageView)findViewById(R.id.LyrPle_Cant2_Play);
    lyrPlePausa1 = (ImageView)findViewById(R.id.LyrPle_Cant1_Pausa);
    lyrPlePausa2 = (ImageView)findViewById(R.id.LyrPle_Cant2_Pausa);

    lyrPlePlay1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            reproductor = MediaPlayer.create(context,
R.raw.lyristes_plebejus_wav1);
            reproductor.seekTo(posicio1);
            posicio1 = 0;
            reproductor.start();
            reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                @Override
                public void onCompletion(MediaPlayer mp) {
                    posicio1 = 0;
                    lyrPlePlay1.setVisibility(View.VISIBLE);
                    lyrPlePausa1.setVisibility(View.GONE);
                }
            });
            lyrPlePlay1.setVisibility(View.GONE);
            lyrPlePausa1.setVisibility(View.VISIBLE);
```

```
            }
        });

        lyrPlePlay2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.lyristes_plebejus_wav2);
                reproductor.seekTo(posicio2);
                posicio2 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio2 = 0;
                        lyrPlePlay2.setVisibility(View.VISIBLE);
                        lyrPlePausa2.setVisibility(View.GONE);
                    }
                });
                lyrPlePlay2.setVisibility(View.GONE);
                lyrPlePausa2.setVisibility(View.VISIBLE);
            }
        });

        lyrPlePausa1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                lyrPlePlay1.setVisibility(View.VISIBLE);
                lyrPlePausa1.setVisibility(View.GONE);
            }
        });

        lyrPlePausa2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio2 = reproductor.getCurrentPosition();
                lyrPlePlay2.setVisibility(View.VISIBLE);
                lyrPlePausa2.setVisibility(View.GONE);
            }
        });
    }
    else if (especie.compareTo("Tettigettalna argentata") == 0) {
        tetArgPlay1 = (ImageView)findViewById(R.id.TetArg_Cant1_Play);
        tetArgPausa1 = (ImageView)findViewById(R.id.TetArg_Cant1_Pausa);

        tetArgPlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tettigettalna_argentata_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        tetArgPlay1.setVisibility(View.VISIBLE);
                        tetArgPausa1.setVisibility(View.GONE);
                    }
                });
                tetArgPlay1.setVisibility(View.GONE);
                tetArgPausa1.setVisibility(View.VISIBLE);
            }
        });

        tetArgPausa1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                tetArgPlay1.setVisibility(View.VISIBLE);
                tetArgPausa1.setVisibility(View.GONE);
            }
        });
    }
    else if (especie.compareTo("Tettigettula pygmea") == 0) {
        tetPygPlay1 = (ImageView)findViewById(R.id.TetPyg_Cant1_Play);
        tetPygPlay2 = (ImageView)findViewById(R.id.TetPyg_Cant2_Play);
        tetPygPausa1 = (ImageView)findViewById(R.id.TetPyg_Cant1_Pausa);
        tetPygPausa2 = (ImageView)findViewById(R.id.TetPyg_Cant2_Pausa);

        tetPygPlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tettigettula_pygmea_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        tetPygPlay1.setVisibility(View.VISIBLE);
                        tetPygPausa1.setVisibility(View.GONE);
                    }
                });
                tetPygPlay1.setVisibility(View.GONE);
                tetPygPausa1.setVisibility(View.VISIBLE);
            }
        });

        tetPygPlay2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tettigettula_pygmea_wav2);
                reproductor.seekTo(posicio2);
                posicio2 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio2 = 0;
                        tetPygPlay2.setVisibility(View.VISIBLE);
                        tetPygPausa2.setVisibility(View.GONE);
                    }
                });
                tetPygPlay2.setVisibility(View.GONE);
                tetPygPausa2.setVisibility(View.VISIBLE);
            }
        });

        tetPygPausa1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                tetPygPlay1.setVisibility(View.VISIBLE);
                tetPygPausa1.setVisibility(View.GONE);
            }
        });

        tetPygPausa2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio2 = reproductor.getCurrentPosition();
                tetPygPlay2.setVisibility(View.VISIBLE);
                tetPygPausa2.setVisibility(View.GONE);
            }
        });
    }
    else if (especie.compareTo("Tibicina corsica fairmairei") == 0) {
        tibCorPlay1 = (ImageView)findViewById(R.id.TibCor_Cant1_Play);
        tibCorPlay2 = (ImageView)findViewById(R.id.TibCor_Cant2_Play);
        tibCorPlay3 = (ImageView)findViewById(R.id.TibCor_Cant3_Play);
        tibCorPausa1 = (ImageView)findViewById(R.id.TibCor_Cant1_Pausa);
        tibCorPausa2 = (ImageView)findViewById(R.id.TibCor_Cant2_Pausa);
        tibCorPausa3 = (ImageView)findViewById(R.id.TibCor_Cant3_Pausa);

        tibCorPlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_corsica_fairmairei_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        tibCorPlay1.setVisibility(View.VISIBLE);
                        tibCorPausa1.setVisibility(View.GONE);
                    }
                });
                tibCorPlay1.setVisibility(View.GONE);
                tibCorPausa1.setVisibility(View.VISIBLE);
            }
        });
```

```
        tibCorPlay2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_corsica_fairmairei_wav2);
                reproductor.seekTo(posicio2);
                posicio2 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio2 = 0;
                        tibCorPlay2.setVisibility(View.VISIBLE);
                        tibCorPausa2.setVisibility(View.GONE);
                    }
                });
                tibCorPlay2.setVisibility(View.GONE);
                tibCorPausa2.setVisibility(View.VISIBLE);
            }
        });

        tibCorPlay3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_corsica_fairmairei_wav3);
                reproductor.seekTo(posicio3);
                posicio3 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio3 = 0;
                        tibCorPlay3.setVisibility(View.VISIBLE);
                        tibCorPausa3.setVisibility(View.GONE);
                    }
                });
                tibCorPlay3.setVisibility(View.GONE);
                tibCorPausa3.setVisibility(View.VISIBLE);
            }
        });

        tibCorPausa1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                tibCorPlay1.setVisibility(View.VISIBLE);
                tibCorPausa1.setVisibility(View.GONE);
            }
        });

        tibCorPausa2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio2 = reproductor.getCurrentPosition();
                tibCorPlay2.setVisibility(View.VISIBLE);
                tibCorPausa2.setVisibility(View.GONE);
            }
        });

        tibCorPausa3.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio3 = reproductor.getCurrentPosition();
                tibCorPlay3.setVisibility(View.VISIBLE);
                tibCorPausa3.setVisibility(View.GONE);
            }
        });
    }
    else if (especie.compareTo("Tibicina garricola") == 0) {
        tibGarPlay1 = (ImageView)findViewById(R.id.TibGar_Cant1_Play);
        tibGarPausa1 = (ImageView)findViewById(R.id.TibGar_Cant1_Pausa);

        tibGarPlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_garricola_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        tibGarPlay1.setVisibility(View.VISIBLE);
                        tibGarPausa1.setVisibility(View.GONE);
                    }
                });
                tibGarPlay1.setVisibility(View.GONE);
                tibGarPausa1.setVisibility(View.VISIBLE);
            }
        });

        tibGarPausa1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                tibGarPlay1.setVisibility(View.VISIBLE);
                tibGarPausa1.setVisibility(View.GONE);
            }
        });
    }
    else if (especie.compareTo("Tibicina haematodes") == 0) {
        tibHaePlay1 = (ImageView)findViewById(R.id.TibHae_Cant1_Play);
        tibHaePausa1 = (ImageView)findViewById(R.id.TibHae_Cant1_Pausa);

        tibHaePlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_haematodes_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        tibHaePlay1.setVisibility(View.VISIBLE);
                        tibHaePausa1.setVisibility(View.GONE);
                    }
                });
                tibHaePlay1.setVisibility(View.GONE);
                tibHaePausa1.setVisibility(View.VISIBLE);
            }
        });

        tibHaePausa1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                tibHaePlay1.setVisibility(View.VISIBLE);
                tibHaePausa1.setVisibility(View.GONE);
            }
        });
    }
    else if (especie.compareTo("Tibicina quadrisignata") == 0) {
        tibQuaPlay1 = (ImageView)findViewById(R.id.TibQua_Cant1_Play);
        tibQuaPlay2 = (ImageView)findViewById(R.id.TibQua_Cant2_Play);
        tibQuaPlay3 = (ImageView)findViewById(R.id.TibQua_Cant3_Play);
        tibQuaPausa1 = (ImageView)findViewById(R.id.TibQua_Cant1_Pausa);
        tibQuaPausa2 = (ImageView)findViewById(R.id.TibQua_Cant2_Pausa);
        tibQuaPausa3 = (ImageView)findViewById(R.id.TibQua_Cant3_Pausa);

        tibQuaPlay1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_quadrisignata_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                    @Override
                    public void onCompletion(MediaPlayer mp) {
                        posicio1 = 0;
                        tibQuaPlay1.setVisibility(View.VISIBLE);
                        tibQuaPausa1.setVisibility(View.GONE);
                    }
                });
                tibQuaPlay1.setVisibility(View.GONE);
                tibQuaPausa1.setVisibility(View.VISIBLE);
            }
        });

        tibQuaPlay2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_quadrisignata_wav2);
                reproductor.seekTo(posicio2);
```

```
                posicio2 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                  @Override
                  public void onCompletion(MediaPlayer mp) {
                    posicio2 = 0;
                    tibQuaPlay2.setVisibility(View.VISIBLE);
                    tibQuaPausa2.setVisibility(View.GONE);
                  }
                });
                tibQuaPlay2.setVisibility(View.GONE);
                tibQuaPausa2.setVisibility(View.VISIBLE);
              }
            });

            tibQuaPlay3.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_quadrisignata_wav3);
                reproductor.seekTo(posicio3);
                posicio3 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                  @Override
                  public void onCompletion(MediaPlayer mp) {
                    posicio3 = 0;
                    tibQuaPlay3.setVisibility(View.VISIBLE);
                    tibQuaPausa3.setVisibility(View.GONE);
                  }
                });
                tibQuaPlay3.setVisibility(View.GONE);
                tibQuaPausa3.setVisibility(View.VISIBLE);
              }
            });

            tibQuaPausa1.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                tibQuaPlay1.setVisibility(View.VISIBLE);
                tibQuaPausa1.setVisibility(View.GONE);
              }
            });

            tibQuaPausa2.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor.pause();
                posicio2 = reproductor.getCurrentPosition();
                tibQuaPlay2.setVisibility(View.VISIBLE);
                tibQuaPausa2.setVisibility(View.GONE);
              }
            });

            tibQuaPausa3.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor.pause();
                posicio3 = reproductor.getCurrentPosition();
                tibQuaPlay3.setVisibility(View.VISIBLE);
                tibQuaPausa3.setVisibility(View.GONE);
              }
            });
          }
          else if (especie.compareTo("Tibicina tomentosa") == 0) {
            tibTomPlay1 = (ImageView)findViewById(R.id.TibTom_Cant1_Play);
            tibTomPlay2 = (ImageView)findViewById(R.id.TibTom_Cant2_Play);
            tibTomPlay3 = (ImageView)findViewById(R.id.TibTom_Cant3_Play);
            tibTomPausa1 = (ImageView)findViewById(R.id.TibTom_Cant1_Pausa);
            tibTomPausa2 = (ImageView)findViewById(R.id.TibTom_Cant2_Pausa);
            tibTomPausa3 = (ImageView)findViewById(R.id.TibTom_Cant3_Pausa);

            tibTomPlay1.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_tomentosa_wav1);
                reproductor.seekTo(posicio1);
                posicio1 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                  @Override
                  public void onCompletion(MediaPlayer mp) {
                    posicio1 = 0;
                    tibTomPlay1.setVisibility(View.VISIBLE);
                    tibTomPausa1.setVisibility(View.GONE);
                  }
                });
                tibTomPlay1.setVisibility(View.GONE);
                tibTomPausa1.setVisibility(View.VISIBLE);
              }
            });

            tibTomPlay2.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_tomentosa_wav2);
                reproductor.seekTo(posicio2);
                posicio2 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                  @Override
                  public void onCompletion(MediaPlayer mp) {
                    posicio2 = 0;
                    tibTomPlay2.setVisibility(View.VISIBLE);
                    tibTomPausa2.setVisibility(View.GONE);
                  }
                });
                tibTomPlay2.setVisibility(View.GONE);
                tibTomPausa2.setVisibility(View.VISIBLE);
              }
            });

            tibTomPlay3.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor = MediaPlayer.create(context,
R.raw.tibicina_tomentosa_wav3);
                reproductor.seekTo(posicio3);
                posicio3 = 0;
                reproductor.start();
                reproductor.setOnCompletionListener(new
MediaPlayer.OnCompletionListener() {
                  @Override
                  public void onCompletion(MediaPlayer mp) {
                    posicio3 = 0;
                    tibTomPlay3.setVisibility(View.VISIBLE);
                    tibTomPausa3.setVisibility(View.GONE);
                  }
                });
                tibTomPlay3.setVisibility(View.GONE);
                tibTomPausa3.setVisibility(View.VISIBLE);
              }
            });

            tibTomPausa1.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor.pause();
                posicio1 = reproductor.getCurrentPosition();
                tibTomPlay1.setVisibility(View.VISIBLE);
                tibTomPausa1.setVisibility(View.GONE);
              }
            });

            tibTomPausa2.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor.pause();
                posicio2 = reproductor.getCurrentPosition();
                tibTomPlay2.setVisibility(View.VISIBLE);
                tibTomPausa2.setVisibility(View.GONE);
              }
            });

            tibTomPausa3.setOnClickListener(new View.OnClickListener() {
              @Override
              public void onClick(View v) {
                reproductor.pause();
                posicio3 = reproductor.getCurrentPosition();
                tibTomPlay3.setVisibility(View.VISIBLE);
                tibTomPausa3.setVisibility(View.GONE);
              }
            });
          }
        }
        else
          finish();
      }

      private void obreFitxaEspecie(String especie) {
        if (especie.compareTo("Cicada barbara lusitanica") == 0) {
          setContentView(R.layout.fitxa_cicada_barbara_lusitanica);
        }
        else if (especie.compareTo("Cicada orni") == 0) {
```

```
            setContentView(R.layout.fitxa_cicada_orni);
        }
        else if (especie.compareTo("Cicadatra atra") == 0) {
            setContentView(R.layout.fitxa_cicadatra_atra);
        }
        else if (especie.compareTo("Cicadetta brevipennis") == 0) {
            setContentView(R.layout.fitxa_cicadetta_brevipennis);
        }
        else if (especie.compareTo("Cicadetta cerdaniensis") == 0) {
            setContentView(R.layout.fitxa_cicadetta_cerdaniensis);
        }
        else if (especie.compareTo("Hilaphura varipes") == 0) {
            setContentView(R.layout.fitxa_hilaphura_varipes);
        }
        else if (especie.compareTo("Lyristes plebejus") == 0) {
            setContentView(R.layout.fitxa_lyristes_plebejus);
        }
        else if (especie.compareTo("Tettigettalna argentata") == 0) {
            setContentView(R.layout.fitxa_tettigettalna_argentata);
        }
        else if (especie.compareTo("Tettigettula pygmea") == 0) {
            setContentView(R.layout.fitxa_tettigettula_pygmea);
        }
        else if (especie.compareTo("Tibicina corsica fairmairei") == 0) {
            setContentView(R.layout.fitxa_tibicina_corsica_fairmairei);
        }
        else if (especie.compareTo("Tibicina garricola") == 0) {
            setContentView(R.layout.fitxa_tibicina_garricola);
        }
        else if (especie.compareTo("Tibicina haematodes") == 0) {
            setContentView(R.layout.fitxa_tibicina_haematodes);
        }
        else if (especie.compareTo("Tibicina quadrisignata") == 0) {
            setContentView(R.layout.fitxa_tibicina_quadrisignata);
        }
        else if (especie.compareTo("Tibicina tomentosa") == 0) {
            setContentView(R.layout.fitxa_tibicina_tomentosa);
        }
    }


    @Override
    public void onClick(View view) {
    }
}
```

Fitxer.java:

```
package cigales.cicadaapp;

/**
 Fitxer és la classe que gestiona les dades de cada fitxer o carpeta que es trobi dins del
directori actual de l'explorador de fitxers. S'utilitza a l'hora
 d'analitzar fitxers enregistrats amb antelació.
 */

public class Fitxer implements Comparable<Fitxer>{
    private String nom;
    private String informacio;
    private String data;
    private String ruta;
    private String imatge;

    public Fitxer(String n, String i, String dt, String r, String im)
    {
        nom = n;
        informacio = i;
        data = dt;
        ruta = r;
        imatge = im;
    }
    public String getNom()
    {
        return nom;
    }
    public String getInformacio()
    {
        return informacio;
    }
    public String getData()
    {
        return data;
    }
    public String getRuta()
    {
        return ruta;
    }
    public String getImatge() {
        return imatge;
    }
    public int compareTo(Fitxer o) {
        if(this.nom != null)
            return this.nom.toLowerCase().compareTo(o.getNom().toLowerCase());
        else
            throw new IllegalArgumentException();
    }
}
```

LlistaEspecies.java:

```
package cigales.cicadaapp;

import android.content.Intent;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;

/**
 LlistaEspecies és la classe que gestiona la pantalla que mostra la llista de totes les
espècies de cigala tractades per l'aplicació.
 */

public class LlistaEspecies extends AppCompatActivity implements
View.OnClickListener {

    RelativeLayout especie1;
    RelativeLayout especie2;
    RelativeLayout especie3;
    RelativeLayout especie4;
    RelativeLayout especie5;
    RelativeLayout especie6;
    RelativeLayout especie7;
    RelativeLayout especie8;
    RelativeLayout especie9;
    RelativeLayout especie10;
    RelativeLayout especie11;
    RelativeLayout especie12;
    RelativeLayout especie13;
    RelativeLayout especie14;

    ImageView imatge1;
    ImageView imatge2;
    ImageView imatge3;
    ImageView imatge4;
    ImageView imatge5;
    ImageView imatge6;
    ImageView imatge7;
    ImageView imatge8;
    ImageView imatge9;
    ImageView imatge10;
    ImageView imatge11;
    ImageView imatge12;
    ImageView imatge13;
    ImageView imatge14;

    TextView text1;
    TextView text1_2;
    TextView text2;
    TextView text3;
    TextView text4;
    TextView text5;
    TextView text6;
    TextView text7;
    TextView text8;
    TextView text9;
    TextView text10;
    TextView text10_2;
    TextView text11;
    TextView text12;
    TextView text13;
    TextView text14;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_llista_especies);

        especie1 = (RelativeLayout)findViewById(R.id.Especie1);
        especie2 = (RelativeLayout)findViewById(R.id.Especie2);
        especie3 = (RelativeLayout)findViewById(R.id.Especie3);
        especie4 = (RelativeLayout)findViewById(R.id.Especie4);
        especie5 = (RelativeLayout)findViewById(R.id.Especie5);
```

```
especie6 = (RelativeLayout)findViewById(R.id.Especie6);
especie7 = (RelativeLayout)findViewById(R.id.Especie7);
especie8 = (RelativeLayout)findViewById(R.id.Especie8);
especie9 = (RelativeLayout)findViewById(R.id.Especie9);
especie10 = (RelativeLayout)findViewById(R.id.Especie10);
especie11 = (RelativeLayout)findViewById(R.id.Especie11);
especie12 = (RelativeLayout)findViewById(R.id.Especie12);
especie13 = (RelativeLayout)findViewById(R.id.Especie13);
especie14 = (RelativeLayout)findViewById(R.id.Especie14);

imatge1 = (ImageView)findViewById(R.id.Imatge1);
imatge2 = (ImageView)findViewById(R.id.Imatge2);
imatge3 = (ImageView)findViewById(R.id.Imatge3);
imatge4 = (ImageView)findViewById(R.id.Imatge4);
imatge5 = (ImageView)findViewById(R.id.Imatge5);
imatge6 = (ImageView)findViewById(R.id.Imatge6);
imatge7 = (ImageView)findViewById(R.id.Imatge7);
imatge8 = (ImageView)findViewById(R.id.Imatge8);
imatge9 = (ImageView)findViewById(R.id.Imatge9);
imatge10 = (ImageView)findViewById(R.id.Imatge10);
imatge11 = (ImageView)findViewById(R.id.Imatge11);
imatge12 = (ImageView)findViewById(R.id.Imatge12);
imatge13 = (ImageView)findViewById(R.id.Imatge13);
imatge14 = (ImageView)findViewById(R.id.Imatge14);

text1 = (TextView)findViewById(R.id.Text1);
text1_2 = (TextView)findViewById(R.id.Text1_2);
text2 = (TextView)findViewById(R.id.Text2);
text3 = (TextView)findViewById(R.id.Text3);
text4 = (TextView)findViewById(R.id.Text4);
text5 = (TextView)findViewById(R.id.Text5);
text6 = (TextView)findViewById(R.id.Text6);
text7 = (TextView)findViewById(R.id.Text7);
text8 = (TextView)findViewById(R.id.Text8);
text9 = (TextView)findViewById(R.id.Text9);
text10 = (TextView)findViewById(R.id.Text10);
text10_2 = (TextView)findViewById(R.id.Text10_2);
text11 = (TextView)findViewById(R.id.Text11);
text12 = (TextView)findViewById(R.id.Text12);
text13 = (TextView)findViewById(R.id.Text13);
text14 = (TextView)findViewById(R.id.Text14);

View.OnClickListener obre1 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(1);
  }
};

View.OnClickListener obre2 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(2);
  }
};

View.OnClickListener obre3 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(3);
  }
};

View.OnClickListener obre4 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(4);
  }
};

View.OnClickListener obre5 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(5);
  }
};

View.OnClickListener obre6 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(6);
  }
};

View.OnClickListener obre7 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(7);
  }
};

View.OnClickListener obre8 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(8);
  }
};

View.OnClickListener obre9 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(9);
  }
};

View.OnClickListener obre10 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(10);
  }
};

View.OnClickListener obre11 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(11);
  }
};

View.OnClickListener obre12 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(12);
  }
};

View.OnClickListener obre13 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(13);
  }
};

View.OnClickListener obre14 = new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    obreEspecie(14);
  }
};

especie1.setOnClickListener(obre1);
especie2.setOnClickListener(obre2);
especie3.setOnClickListener(obre3);
especie4.setOnClickListener(obre4);
especie5.setOnClickListener(obre5);
especie6.setOnClickListener(obre6);
especie7.setOnClickListener(obre7);
especie8.setOnClickListener(obre8);
especie9.setOnClickListener(obre9);
especie10.setOnClickListener(obre10);
especie11.setOnClickListener(obre11);
especie12.setOnClickListener(obre12);
especie13.setOnClickListener(obre13);
especie14.setOnClickListener(obre14);

imatge1.setOnClickListener(obre1);
imatge2.setOnClickListener(obre2);
imatge3.setOnClickListener(obre3);
imatge4.setOnClickListener(obre4);
imatge5.setOnClickListener(obre5);
imatge6.setOnClickListener(obre6);
imatge7.setOnClickListener(obre7);
imatge8.setOnClickListener(obre8);
imatge9.setOnClickListener(obre9);
imatge10.setOnClickListener(obre10);
imatge11.setOnClickListener(obre11);
imatge12.setOnClickListener(obre12);
imatge13.setOnClickListener(obre13);
imatge14.setOnClickListener(obre14);

text1.setOnClickListener(obre1);
text1_2.setOnClickListener(obre1);
text2.setOnClickListener(obre2);
text3.setOnClickListener(obre3);
text4.setOnClickListener(obre4);
text5.setOnClickListener(obre5);
text6.setOnClickListener(obre6);
text7.setOnClickListener(obre7);
text8.setOnClickListener(obre8);
text9.setOnClickListener(obre9);
text10.setOnClickListener(obre10);
text10_2.setOnClickListener(obre10);
```

```
        text11.setOnClickListener(obre11);
        text12.setOnClickListener(obre12);
        text13.setOnClickListener(obre13);
        text14.setOnClickListener(obre14);
    }

    @Override
    public void onClick(View view) {
    }

    public void obreEspecie(int especie) {
        Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
        Bundle b = new Bundle();

        if (especie == 1) {
            b.putString("especie", "Cicada barbara lusitanica");
        }
        else if (especie == 2) {
            b.putString("especie", "Cicada orni");
        }
        else if (especie == 3) {
            b.putString("especie", "Cicadatra atra");
        }
        else if (especie == 4) {
            b.putString("especie", "Cicadetta brevipennis");
        }
        else if (especie == 5) {
            b.putString("especie", "Cicadetta cerdaniensis");
        }
        else if (especie == 6) {
            b.putString("especie", "Hilaphura varipes");
        }
        else if (especie == 7) {
            b.putString("especie", "Lyristes plebejus");
        }
        else if (especie == 8) {
            b.putString("especie", "Tettigettalna argentata");
        }
        else if (especie == 9) {
            b.putString("especie", "Tettigettula pygmea");
        }
        else if (especie == 10) {
            b.putString("especie", "Tibicina corsica fairmairei");
        }
        else if (especie == 11) {
            b.putString("especie", "Tibicina garricola");
        }
        else if (especie == 12) {
            b.putString("especie", "Tibicina haematodes");
        }
        else if (especie == 13) {
            b.putString("especie", "Tibicina quadrisignata");
        }
        else if (especie == 14) {
            b.putString("especie", "Tibicina tomentosa");
        }

        intent.putExtras(b);
        startActivity(intent);
    }
}
```

Resultats.java:

```
package cigales.cicadaapp;

import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;

import java.util.ArrayList;

/**
 Resultats és la classe que gestiona la pantalla que mostra els resultats de l'anàlisi d'un
cant.
 */

public class Resultats extends AppCompatActivity implements View.OnClickListener {

    RelativeLayout segonaFila;
    RelativeLayout terceraFila;
    RelativeLayout quartaFila;
    RelativeLayout cinquenaFila;
    RelativeLayout sisenaFila;

    ImageView CigalaCercle;
    ImageView CigalaCercle2;
    ImageView CigalaCercle3;
    ImageView CigalaCercle4;

    TextView NomCigala;
    TextView NomCigala2;
    TextView NomCigala3;
    TextView NomCigala4;

    TextView PercentatgeSemblanca;
    TextView PercentatgeSemblanca2;
    TextView PercentatgeSemblanca3;
    TextView PercentatgeSemblanca4;

    String especie1;
    String especie2;
    String especie3;
    String especie4;


    TextView FrequenciaMostreigExigua;

    Context context = this;

    boolean mostraSisenaFila;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_resultats);


        //Segona fila
        segonaFila = (RelativeLayout)findViewById(R.id.SegonaFila);
        segonaFila.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
                Bundle b = new Bundle();
                b.putString("especie", especie1);
                intent.putExtras(b);
                startActivity(intent);
            }
        });

        //Tercera fila
        terceraFila = (RelativeLayout)findViewById(R.id.TerceraFila);
        terceraFila.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
                Bundle b = new Bundle();
                b.putString("especie", especie2);
                intent.putExtras(b);
                startActivity(intent);
            }
        });

        //Quarta fila
        quartaFila = (RelativeLayout)findViewById(R.id.QuartaFila);
        quartaFila.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
                Bundle b = new Bundle();
                b.putString("especie", especie3);
                intent.putExtras(b);
                startActivity(intent);
            }
        });

        //Cinquena fila
        cinquenaFila = (RelativeLayout)findViewById(R.id.CinquenaFila);
        cinquenaFila.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
                Bundle b = new Bundle();
                b.putString("especie", especie4);
                intent.putExtras(b);
                startActivity(intent);
            }
        });

        //Sisena fila
        sisenaFila = (RelativeLayout)findViewById(R.id.SisenaFila);
```

```
sisenaFila.setOnClickListener(this);


//Primer cercle
CigalaCercle = (ImageView)findViewById(R.id.CigalaCercle);
CigalaCercle.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie1);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Segon cercle
CigalaCercle2 = (ImageView)findViewById(R.id.CigalaCercle2);
CigalaCercle2.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie2);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Tercer cercle
CigalaCercle3 = (ImageView)findViewById(R.id.CigalaCercle3);
CigalaCercle3.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie3);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Quart cercle
CigalaCercle4 = (ImageView)findViewById(R.id.CigalaCercle4);
CigalaCercle4.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie4);
    intent.putExtras(b);
    startActivity(intent);
  }
});


//Primer nom
NomCigala = (TextView)findViewById(R.id.NomCigala);
NomCigala.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie1);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Segon nom
NomCigala2 = (TextView)findViewById(R.id.NomCigala2);
NomCigala2.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie2);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Tercer nom
NomCigala3 = (TextView)findViewById(R.id.NomCigala3);
NomCigala3.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie3);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Quart nom
NomCigala4 = (TextView)findViewById(R.id.NomCigala4);
NomCigala4.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie4);
    intent.putExtras(b);
    startActivity(intent);
  }
});


//Primer percentatge
PercentatgeSemblanca = (TextView)findViewById(R.id.PercentatgeSemblanca);
PercentatgeSemblanca.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie1);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Segon percentatge
PercentatgeSemblanca2 = (TextView)findViewById(R.id.PercentatgeSemblanca2);
PercentatgeSemblanca2.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie2);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Tercer percentatge
PercentatgeSemblanca3 = (TextView)findViewById(R.id.PercentatgeSemblanca3);
PercentatgeSemblanca3.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie3);
    intent.putExtras(b);
    startActivity(intent);
  }
});

//Quart percentatge
PercentatgeSemblanca4 = (TextView)findViewById(R.id.PercentatgeSemblanca4);
PercentatgeSemblanca4.setOnClickListener(new View.OnClickListener() {
  @Override
  public void onClick(View v) {
    Intent intent = new Intent("cigales.cicadaapp.FitxaEspecie");
    Bundle b = new Bundle();
    b.putString("especie", especie4);
    intent.putExtras(b);
    startActivity(intent);
  }
});


    FrequenciaMostreigExigua                                    =
(TextView)findViewById(R.id.FrequenciaMostreigExigua);

    //Mostra les espècies, amb els seus corresponents coeficients de semblança, amb
les quals s'ha inicialitzat l'objecte Resultats actual.
    Bundle b = getIntent().getExtras();
    ArrayList<String> especies = new ArrayList<>();
    if(b != null)
      especies = b.getStringArrayList("llistaEspecies");

    mostraSisenaFila = false;
    mostraEspeciesTrobades(especies);
  }

  private void mostraEspeciesTrobades(ArrayList<String> especies) {
    if (especies.size() >= 4) {
      segonaFila.setVisibility(View.VISIBLE);
      terceraFila.setVisibility(View.VISIBLE);
      quartaFila.setVisibility(View.VISIBLE);
      cinquenaFila.setVisibility(View.VISIBLE);
```

```
        assignaImatgePercentatgeIText(2, especies.get(0));
        assignaImatgePercentatgeIText(3, especies.get(1));
        assignaImatgePercentatgeIText(4, especies.get(2));
        assignaImatgePercentatgeIText(5, especies.get(3));
    }
    else if (especies.size() == 3) {
        segonaFila.setVisibility(View.VISIBLE);
        terceraFila.setVisibility(View.VISIBLE);
        quartaFila.setVisibility(View.VISIBLE);
        cinquenaFila.setVisibility(View.GONE);

        assignaImatgePercentatgeIText(2, especies.get(0));
        assignaImatgePercentatgeIText(3, especies.get(1));
        assignaImatgePercentatgeIText(4, especies.get(2));
        assignaEspecie(5, "");
    }
    else if (especies.size() == 2) {
        segonaFila.setVisibility(View.VISIBLE);
        terceraFila.setVisibility(View.VISIBLE);
        quartaFila.setVisibility(View.GONE);
        cinquenaFila.setVisibility(View.GONE);

        assignaImatgePercentatgeIText(2, especies.get(0));
        assignaImatgePercentatgeIText(3, especies.get(1));
        assignaEspecie(4, "");
        assignaEspecie(5, "");
    }
    else if (especies.size() == 1) {
        segonaFila.setVisibility(View.VISIBLE);
        terceraFila.setVisibility(View.GONE);
        quartaFila.setVisibility(View.GONE);
        cinquenaFila.setVisibility(View.GONE);

        assignaImatgePercentatgeIText(2, especies.get(0));
        assignaEspecie(3, "");
        assignaEspecie(4, "");
        assignaEspecie(5, "");
    }
    else {
        segonaFila.setVisibility(View.GONE);
        terceraFila.setVisibility(View.GONE);
        quartaFila.setVisibility(View.GONE);
        cinquenaFila.setVisibility(View.GONE);

        assignaEspecie(2, "");
        assignaEspecie(3, "");
        assignaEspecie(4, "");
        assignaEspecie(5, "");
    }
  }

  private void assignaEspecie(int fila, String especie) {
    if (fila == 2) {
        especie1 = especie;
    }
    else if (fila == 3) {
        especie2 = especie;
    }
    else if (fila == 4) {
        especie3 = especie;
    }
    else if (fila == 5) {
        especie4 = especie;
    }
  }

  //Tria una imatge, un text i un percentatge per a cada fila que s'ha de mostrar a partir de l'String entrat.
  private void assignaImatgePercentatgeIText(int fila, String s) {

      if (s.length() > 25 && s.substring(0, 25).compareTo("Cicada barbara lusitanica") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(25) != '*' ? "Cicada barbara lusitanica**" : "Cicada barbara lusitanica*"), (int) Math.round(Double.valueOf(s.substring(26, s.length()))));
          mostraSisenaFila = true;

FrequenciaMostreigExigua.setText(context.getString(R.string.AdvertenciaCicadaBarbaraLusitanica));
          assignaImatge(fila, R.drawable.cicada_barbara_lusitanica_cercle);
          assignaEspecie(fila, "Cicada barbara lusitanica");
      }
      else if (s.length() > 11 && s.substring(0, 11).compareTo("Cicada orni") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(11) != '*' ? "Cicada orni" : "Cicada orni*"), (int) Math.round(Double.valueOf(s.substring(12, s.length()))));
          if (s.charAt(11) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.cicada_orni_cercle);
          assignaEspecie(fila, "Cicada orni");
      }
      else if (s.length() > 14 && s.substring(0, 14).compareTo("Cicadatra atra") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(14) != '*' ? "Cicadatra atra" : "Cicadatra atra*"), (int) Math.round(Double.valueOf(s.substring(15, s.length()))));
          if (s.charAt(14) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.cicadatra_atra_cercle);
          assignaEspecie(fila, "Cicadatra atra");
      }
      else if (s.length() > 21 && s.substring(0, 21).compareTo("Cicadetta brevipennis") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(21) != '*' ? "Cicadetta brevipennis" : "Cicadetta brevipennis*"), (int) Math.round(Double.valueOf(s.substring(22, s.length()))));
          if (s.charAt(21) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.cicadetta_brevipennis_cercle);
          assignaEspecie(fila, "Cicadetta brevipennis");
      }
      else if (s.length() > 22 && s.substring(0, 22).compareTo("Cicadetta cerdaniensis") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(22) != '*' ? "Cicadetta cerdaniensis" : "Cicadetta cerdaniensis*"), (int) Math.round(Double.valueOf(s.substring(23, s.length()))));
          if (s.charAt(22) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.cicadetta_cerdaniensis_cercle);
          assignaEspecie(fila, "Cicadetta cerdaniensis");
      }
      else if (s.length() > 17 && s.substring(0, 17).compareTo("Hilaphura varipes") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(17) != '*' ? "Hilaphura varipes" : "Hilaphura varipes*"), (int) Math.round(Double.valueOf(s.substring(18, s.length()))));
          if (s.charAt(17) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.hilaphura_varipes_cercle);
          assignaEspecie(fila, "Hilaphura varipes");
      }
      else if (s.length() > 17 && s.substring(0, 17).compareTo("Lyristes plebejus") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(17) != '*' ? "Lyristes plebejus" : "Lyristes plebejus*"), (int) Math.round(Double.valueOf(s.substring(18, s.length()))));
          if (s.charAt(17) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.lyristes_plebejus_cercle);
          assignaEspecie(fila, "Lyristes plebejus");
      }
      else if (s.length() > 23 && s.substring(0, 23).compareTo("Tettigettalna argentata") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(23) != '*' ? "Tettigettalna argentata" : "Tettigettalna argentata*"), (int) Math.round(Double.valueOf(s.substring(24, s.length()))));
          if (s.charAt(23) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.tettigettalna_argentata_cercle);
          assignaEspecie(fila, "Tettigettalna argentata");
      }
      else if (s.length() > 19 && s.substring(0, 19).compareTo("Tettigettula pygmea") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(19) != '*' ? "Tettigettula pygmea" : "Tettigettula pygmea*"), (int) Math.round(Double.valueOf(s.substring(20, s.length()))));
          if (s.charAt(19) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.tettigettula_pygmea_cercle);
          assignaEspecie(fila, "Tettigettula pygmea");
      }
      else if (s.length() > 27 && s.substring(0, 27).compareTo("Tibicina corsica fairmairei") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(27) != '*' ? "Tibicina corsica fairmairei" : "Tibicina corsica fairmairei*"), (int) Math.round(Double.valueOf(s.substring(28, s.length()))));
          if (s.charAt(27) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.tibicina_corsica_fairmairei_cercle);
          assignaEspecie(fila, "Tibicina corsica fairmairei");
      }
      else if (s.length() > 18 && s.substring(0, 18).compareTo("Tibicina garricola") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(18) != '*' ? "Tibicina garricola" : "Tibicina garricola*"), (int) Math.round(Double.valueOf(s.substring(19, s.length()))));
          if (s.charAt(18) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.tibicina_garricola_cercle);
          assignaEspecie(fila, "Tibicina garricola");
      }
      else if (s.length() > 19 && s.substring(0, 19).compareTo("Tibicina haematodes") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(19) != '*' ? "Tibicina haematodes" : "Tibicina haematodes*"), (int) Math.round(Double.valueOf(s.substring(20, s.length()))));
          if (s.charAt(19) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.tibicina_haematodes_cercle);
          assignaEspecie(fila, "Tibicina haematodes");
      }
      else if (s.length() > 22 && s.substring(0, 22).compareTo("Tibicina quadrisignata") == 0) {
          assignaPercentatgeIText(fila, (s.charAt(22) != '*' ? "Tibicina quadrisignata" : "Tibicina quadrisignata*"), (int) Math.round(Double.valueOf(s.substring(23, s.length()))));
          if (s.charAt(22) == '*') mostraSisenaFila = true;
          assignaImatge(fila, R.drawable.tibicina_quadrisignata_cercle);
          assignaEspecie(fila, "Tibicina quadrisignata");
```

```
        }
        else if (s.length() > 18 && s.substring(0, 18).compareTo("Tibicina tomentosa") ==
0) {
            assignaPercentatgeIText(fila, (s.charAt(18) != '*' ? "Tibicina tomentosa" :
"Tibicina tomentosa*"), (int) Math.round(Double.valueOf(s.substring(19, s.length()))));
            if (s.charAt(18) == '*') mostraSisenaFila = true;
            assignaImatge(fila, R.drawable.tibicina_tomentosa_cercle);
            assignaEspecie(fila, "Tibicina tomentosa");
        }

        if (mostraSisenaFila) sisenaFila.setVisibility(View.VISIBLE);
        else sisenaFila.setVisibility(View.GONE);
    }

    private void assignaImatge(int fila, int cercle) {

        if (fila == 2) {
            CigalaCercle.setImageDrawable(ContextCompat.getDrawable(context, cercle));
        }
        else if (fila == 3) {
            CigalaCercle2.setImageDrawable(ContextCompat.getDrawable(context,
cercle));
        }
        else if (fila == 4) {
            CigalaCercle3.setImageDrawable(ContextCompat.getDrawable(context,
cercle));
        }
        else if (fila == 5) {
            CigalaCercle4.setImageDrawable(ContextCompat.getDrawable(context,
cercle));
        }
    }

    private void assignaPercentatgeIText(int fila, String s, int percentatge) {
        if (fila == 2) {
            NomCigala.setText(s);
            PercentatgeSemblanca.setText(String.valueOf(percentatge) + "%");
        }
        else if (fila == 3) {
            NomCigala2.setText(s);
            PercentatgeSemblanca2.setText(String.valueOf(percentatge) + "%");
        }
        else if (fila == 4) {
            NomCigala3.setText(s);
            PercentatgeSemblanca3.setText(String.valueOf(percentatge) + "%");
        }
        else if (fila == 5) {
            NomCigala4.setText(s);
            PercentatgeSemblanca4.setText(String.valueOf(percentatge) + "%");
        }
    }


    @Override
    public void onClick(View view) {
    }
}
```

parpelleig.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <alpha android:fromAlpha="1.0"
        android:toAlpha="0.0"
        android:interpolator="@android:anim/accelerate_interpolator"
        android:duration="1800"
        android:repeatMode="reverse"
        android:repeatCount="infinite"/>
</set>
```

rotacio.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
    <rotate android:fromDegrees="0"
        android:toDegrees="360"
        android:pivotX="50%"
        android:pivotY="50%"
        android:duration="6000"
        android:repeatMode="restart"
        android:repeatCount="infinite"
        android:interpolator="@android:anim/cycle_interpolator"/>

</set>
```

activity_activitat_principal.xml:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:paddingLeft="10dp"
    android:paddingRight="10dp"
    android:paddingTop="10dp"
    android:paddingBottom="@dimen/activity_vertical_margin"
tools:context=".ActivitatPrincipal"
    android:background="@drawable/taygetos"
    android:clickable="true"
    android:id="@+id/disseny">

    <Button
        android:layout_width="wrap_content"
        android:layout_height="40dp"
        android:text="@string/LesEspecies"
        android:id="@+id/LlistaEspecies"
        android:width="250dp"
        android:height="30dp"
        android:layout_above="@+id/Gravacio"
        android:layout_centerHorizontal="true"
        android:background="#e6f8fcff"
        android:textColor="#000000" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="40dp"
        android:text="@string/IniciaUnaGravacio"
        android:id="@+id/Gravacio"
        android:width="250dp"
        android:height="30dp"
        android:layout_above="@+id/AnalisiFitxer"
        android:layout_marginTop="10dp"
        android:layout_centerHorizontal="true"
        android:background="#e6f8fcff"
        android:textColor="#000000" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="40dp"
        android:text="@string/AnalitzaUnFitxer"
        android:id="@+id/AnalisiFitxer"
        android:width="250dp"
        android:height="30dp"
        android:layout_alignParentBottom="true"
        android:layout_marginTop="10dp"
        android:layout_centerHorizontal="true"
        android:background="#e6f8fcff"
        android:textColor="#000000" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="30dp"
        android:drawableLeft="@drawable/catala_mini_espai"
        style="?android:attr/borderlessButtonStyle"
        android:background="@null"
        android:id="@+id/Idioma"
        android:height="25dp"
        android:textAlignment="viewEnd"
        android:layout_gravity="end"
        android:text="@string/Catala"
        android:onClick="mostraIdiomes"
        android:paddingRight="50dp"
        android:layout_alignTop="@+id/Configuracio"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"></Button>

    <ImageView
        android:id="@+id/Configuracio"
        android:layout_width="25dp"
        android:layout_height="28dp"
        android:src="@drawable/configuracio"
        android:layout_alignParentTop="true"
        android:layout_alignParentRight="true"
        android:layout_alignParentEnd="true"
        android:onClick="mostraMenuConfiguracio"/>

    <ImageView
        android:id="@+id/logotip"
        android:layout_width="125dp"
        android:layout_height="125dp"
        android:src="@drawable/logotip_cercle"
        android:layout_marginTop="10dp"
        android:layout_below="@+id/Idioma"
        android:layout_centerHorizontal="true" />
```

```
    <ImageView
        android:id="@+id/titol"
        android:layout_width="181dp"
        android:layout_height="58dp"
        android:src="@drawable/titol_reduit"
        android:layout_below="@+id/logotip"
        android:layout_marginTop="0dp"
        android:layout_centerHorizontal="true" />

    <ListView
        android:layout_width="130dp"
        android:layout_height="125dp"
        android:id="@+id/LlistaIdiomes"
        android:layout_below="@+id/Idioma"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:visibility="gone"
        android:background="#bed8ff" />

    <ListView
        android:layout_width="300dp"
        android:layout_height="68dp"
        android:id="@+id/LlistaConfiguracio"
        android:layout_below="@+id/Configuracio"
        android:layout_alignParentRight="true"
        android:visibility="gone"
        android:background="#bed8ff" />


</RelativeLayout>
```

activity_enregistrador_audio.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
    android:layout_height="match_parent" android:paddingLeft="10dp"
    android:paddingRight="10dp"
    android:paddingTop="10dp"
    android:paddingBottom="@dimen/activity_vertical_margin"
tools:context="cigales.cicadaapp.EnregistradorAudio"
    android:background="@drawable/blancverd_degradat"
    android:clickable="true"
    android:id="@+id/dissenyPantallaGravacio">

    <ImageView
        android:id="@+id/Carregant"
        android:layout_width="180dp"
        android:layout_height="180dp"
        android:src="@drawable/carregant_blanc"
        android:layout_gravity="center"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:visibility="gone"/>

    <ImageView
        android:id="@+id/Microfon"
        android:layout_width="110dp"
        android:layout_height="110dp"
        android:src="@drawable/microfon"
        android:layout_gravity="center"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:clickable="true"
        android:visibility="gone"/>

    <ImageView
        android:id="@+id/RellotgeSorra"
        android:layout_width="110dp"
        android:layout_height="110dp"
        android:src="@drawable/rellotge_de_sorra"
        android:layout_gravity="center"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:clickable="true"
        android:visibility="gone"/>

    <ImageView
        android:id="@+id/MicrofonFonsBlanc"
        android:layout_width="110dp"
        android:layout_height="110dp"
        android:src="@drawable/microfon_fons_blanc"
        android:layout_gravity="center"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:clickable="true"/>

    <ImageView
        android:id="@+id/Reproduccio"
        android:layout_width="110dp"
        android:layout_height="110dp"
        android:src="@drawable/reproduccio"
        android:layout_gravity="center"
        android:layout_centerVertical="true"
        android:layout_centerHorizontal="true"
        android:clickable="true"
        android:visibility="gone"/>

    <TextView android:text="@string/title_activity_enregistrador_audio"
android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:layout_marginTop="48dp"
        android:typeface="normal"
        android:textStyle="bold"
        android:textSize="20dp"
        android:layout_centerHorizontal="true"
        android:visibility="gone"
        android:id="@+id/GravacioEnCurs"/>

    <TextView android:text="@string/Analitzant"
android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:layout_marginTop="48dp"
        android:typeface="normal"
        android:textStyle="bold"
        android:textSize="20dp"
        android:layout_centerHorizontal="true"
        android:visibility="gone"
        android:id="@+id/Analitzant"/>

    <TextView android:text="@string/ReproduccioEnCurs"
android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:layout_marginTop="48dp"
        android:typeface="normal"
        android:textStyle="bold"
        android:textSize="20dp"
        android:layout_centerHorizontal="true"
        android:visibility="gone"
        android:id="@+id/ReproduccioEnCurs"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="40dp"
        android:text="@string/Reprodueix"
        android:id="@+id/Reprodueix"
        android:width="250dp"
        android:height="30dp"
        android:layout_above="@+id/Analitza"
        android:layout_centerHorizontal="true"
        android:background="#e6f8fcff"
        android:textColor="#000000"
        android:visibility="gone"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="40dp"
        android:text="@string/Atura"
        android:id="@+id/Atura"
        android:width="250dp"
        android:height="30dp"
        android:layout_below="@+id/Reproduccio"
        android:layout_marginTop="57dp"
        android:layout_centerHorizontal="true"
        android:background="#e6f8fcff"
        android:textColor="#000000"
        android:visibility="gone"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="40dp"
        android:text="@string/Analitza"
        android:id="@+id/Analitza"
        android:width="250dp"
        android:height="30dp"
        android:layout_above="@+id/Descarta"
        android:layout_marginTop="10dp"
        android:layout_centerHorizontal="true"
        android:background="#e6f8fcff"
        android:textColor="#000000"
        android:visibility="gone"/>

    <Button
        android:layout_width="wrap_content"
        android:layout_height="40dp"
```

```
    android:text="@string/Descarta"
    android:id="@+id/Descarta"
    android:width="250dp"
    android:height="30dp"
    android:layout_alignParentBottom="true"
    android:layout_marginTop="10dp"
    android:layout_centerHorizontal="true"
    android:background="#e6f8fcff"
    android:textColor="#000000"
    android:visibility="gone"/>

</RelativeLayout>
```

activity_llista_especies.xml:

```
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff"
  tools:context="cigales.cicadaapp.LlistaEspecies">



  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/LlistaEspecies_Layout">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:tools="http://schemas.android.com/tools"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/Titol_LlistaEspecies"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/LlistaEspeciesCurt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/TextLlistaEspecies"
        android:layout_centerVertical="true"
        android:gravity="center_horizontal"
        android:layout_centerHorizontal="true"
        android:textSize="26sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:tools="http://schemas.android.com/tools"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/Especie1"
      android:layout_marginTop="5dp"
      android:layout_marginLeft="5dp"
      android:layout_marginRight="5dp"
      android:background="@drawable/degradat_blanc"
      android:layout_below="@id/Titol_LlistaEspecies">

      <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge1"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/cicada_barbara_lusitanica_370"/>

      <TextView android:text="@string/CicadaBarbara"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text1"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge1"
        android:layout_marginLeft="120dp"
        android:layout_marginTop="12dp"
        android:layout_alignTop="@+id/Imatge1"
        android:textSize="20sp"/>

      <TextView android:text="@string/SspLusitanica"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text1_2"
        android:textSize="16sp"
        android:layout_below="@+id/Text1"
        android:layout_alignLeft="@+id/Text1"
        android:layout_marginTop="0dp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:tools="http://schemas.android.com/tools"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/Especie2"
      android:layout_marginTop="5dp"
      android:layout_marginLeft="5dp"
      android:layout_marginRight="5dp"
      android:background="@drawable/degradat_blanc"
      android:layout_below="@id/Especie1">

      <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge2"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/cicada_orni_370"/>

      <TextView android:text="@string/CicadaOrni"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text2"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge2"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:tools="http://schemas.android.com/tools"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/Especie3"
      android:layout_marginTop="5dp"
      android:layout_marginLeft="5dp"
      android:layout_marginRight="5dp"
      android:background="@drawable/degradat_blanc"
      android:layout_below="@id/Especie2">

      <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge3"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/cicadatra_atra_370"/>

      <TextView android:text="@string/CicadatraAtra"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text3"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge3"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:tools="http://schemas.android.com/tools"
      android:layout_width="match_parent"
```

```
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie4"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie3">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge4"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/cicadetta_brevipennis_370"/>

    <TextView android:text="@string/CicadettaBrevipennis"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text4"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge4"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie5"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie4">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge5"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/cicadetta_cerdaniensis_370"/>

    <TextView android:text="@string/CicadettaCerdaniensis"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text5"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge5"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie6"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie5">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge6"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/hilaphura_varipes_370"/>

    <TextView android:text="@string/HilaphuraVaripes"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text6"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge6"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie7"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie6">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge7"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/lyristes_plebejus_370"/>

    <TextView android:text="@string/LyristesPlebejus"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text7"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge7"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie8"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie7">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge8"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/tettigettalna_argentata_370"/>

    <TextView android:text="@string/TettigettalnaArgentata"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text8"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge8"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie9"
    android:layout_marginTop="5dp"
```

```
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie8">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge9"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/tettigettula_pygmea_370"/>

    <TextView android:text="@string/TettigettulaPygmea"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text9"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge9"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie10"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie9">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge10"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/tibicina_corsica_fairmairei_370"/>

    <TextView android:text="@string/TibicinaCorsica"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text10"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge10"
        android:layout_marginLeft="120dp"
        android:layout_marginTop="12dp"
        android:layout_alignTop="@+id/Imatge10"
        android:textSize="20sp"/>

    <TextView android:text="@string/SspFairmairei"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text10_2"
        android:textSize="16sp"
        android:layout_below="@+id/Text10"
        android:layout_alignLeft="@+id/Text10"
        android:layout_marginTop="0dp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie11"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie10">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge11"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/tibicina_garricola_370"/>

    <TextView android:text="@string/TibicinaGarricola"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text11"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge11"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie12"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie11">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge12"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/tibicina_haematodes_370"/>

    <TextView android:text="@string/TibicinaHaematodes"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text12"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge12"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/Especie13"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="5dp"
    android:background="@drawable/degradat_blanc"
    android:layout_below="@id/Especie12">

    <ImageView
        android:layout_width="100dp"
        android:layout_height="75dp"
        android:id="@+id/Imatge13"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:src="@drawable/tibicina_quadrisignata_370"/>

    <TextView android:text="@string/TibicinaQuadrisignata"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/Text13"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/Imatge13"
        android:layout_marginLeft="120dp"
        android:textSize="20sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
```

```
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="0dp"
        android:paddingBottom="0dp"
        android:id="@+id/Especie14"
        android:layout_marginTop="5dp"
        android:layout_marginLeft="5dp"
        android:layout_marginRight="5dp"
        android:background="@drawable/degradat_blanc"
        android:layout_below="@id/Especie13">

        <ImageView
          android:layout_width="100dp"
          android:layout_height="75dp"
          android:id="@+id/Imatge14"
          android:layout_alignParentTop="true"
          android:layout_alignParentLeft="true"
          android:layout_alignParentStart="true"
          android:src="@drawable/tibicina_tomentosa_370"/>

        <TextView android:text="@string/TibicinaTomentosa"
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:textColor="#000000"
          android:id="@+id/Text14"
          android:layout_centerVertical="true"
          android:layout_alignLeft="@+id/Imatge14"
          android:layout_marginLeft="120dp"
          android:textSize="20sp"/>
      </RelativeLayout>

      <TextView android:text=""
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/LlistaEspecies_EspaiBlanc"
        android:layout_centerVertical="true"
        android:layout_below="@+id/Especie14"
        android:layout_marginLeft="120dp"
        android:textSize="5sp"/>
    </RelativeLayout>
  </ScrollView>
```

activity_resultats.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
    tools:context="cigales.cicadaapp.Resultats">


    <RelativeLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="60dp"
        android:layout_alignParentTop="true"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:background="@drawable/degradat_blau"
        android:id="@+id/PrimeraFila">

        <TextView
          android:layout_width="wrap_content"
          android:layout_height="wrap_content"
          android:textAppearance="?android:attr/textAppearanceLarge"
          android:text="@string/Resultats"
          android:id="@+id/Resultats"
          android:layout_centerVertical="true"
          android:layout_alignParentLeft="true"
          android:layout_alignParentStart="true"
          android:layout_marginLeft="26dp"
          android:textSize="32sp"
          android:textColor="#ffffff" />
    </RelativeLayout>


    <RelativeLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="90dp"
        android:layout_below="@+id/PrimeraFila"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:id="@+id/SegonaFila"
        android:background="@drawable/degradat_blanc">

        <ImageView
          android:id="@+id/CigalaCercle"
          android:layout_width="65dp"
          android:layout_height="65dp"
          android:src="@drawable/cicada_orni_cercle"
          android:layout_gravity="center"
          android:visibility="visible"
          android:layout_centerVertical="true"
          android:layout_marginLeft="20dp"/>

        <TextView
          android:layout_width="130dp"
          android:layout_height="40dp"
          android:textAppearance="?android:attr/textAppearanceLarge"
          android:text="@string/CicadaOrni"
          android:id="@+id/NomCigala"
          android:textColor="#010101"
          android:visibility="visible"
          android:gravity="left|center"
          android:textSize="18sp"
          android:layout_alignTop="@+id/CigalaCercle"
          android:layout_toRightOf="@id/CigalaCercle"
          android:layout_alignBottom="@+id/CigalaCercle"
          android:layout_marginLeft="20dp"
          android:textStyle="italic" />

        <TextView
          android:layout_width="wrap_content"
          android:layout_height="40dp"
          android:textAppearance="?android:attr/textAppearanceLarge"
          android:text="x%"
          android:id="@+id/PercentatgeSemblanca"
          android:textColor="#010101"
          android:visibility="visible"
          android:textStyle="bold"
          android:textSize="24sp"
          android:gravity="center_vertical|center_horizontal"
          android:layout_alignBottom="@+id/NomCigala"
          android:layout_alignParentRight="true"
          android:layout_alignParentEnd="true"
          android:layout_marginRight="25dp"
          android:layout_marginEnd="25dp"
          android:layout_alignTop="@+id/NomCigala" />
    </RelativeLayout>

    <RelativeLayout
        android:orientation="vertical"
        android:layout_width="match_parent"
        android:layout_height="90dp"
        android:layout_below="@+id/SegonaFila"
        android:layout_alignParentLeft="true"
        android:layout_alignParentStart="true"
        android:id="@+id/TerceraFila"
        android:background="@drawable/degradat_blanc">


        <ImageView
          android:id="@+id/CigalaCercle2"
          android:layout_width="65dp"
          android:layout_height="65dp"
          android:src="@drawable/cicadetta_brevipennis_cercle"
          android:layout_gravity="center"
          android:visibility="visible"
          android:layout_centerVertical="true"
          android:layout_marginLeft="20dp"/>

        <TextView
          android:layout_width="130dp"
          android:layout_height="40dp"
          android:textAppearance="?android:attr/textAppearanceLarge"
          android:text="@string/CicadettaBrevipennis"
          android:id="@+id/NomCigala2"
          android:textColor="#010101"
          android:visibility="visible"
          android:gravity="left|center"
          android:textSize="18sp"
          android:layout_alignTop="@+id/CigalaCercle2"
          android:layout_toRightOf="@id/CigalaCercle2"
          android:layout_alignBottom="@+id/CigalaCercle2"
          android:layout_marginLeft="20dp"
          android:textStyle="italic" />

        <TextView
          android:layout_width="wrap_content"
          android:layout_height="40dp"
          android:textAppearance="?android:attr/textAppearanceLarge"
          android:text="y%"
          android:id="@+id/PercentatgeSemblanca2"
          android:textColor="#010101"
          android:visibility="visible"
```

```
    android:textStyle="bold"
    android:textSize="24sp"
    android:gravity="center_vertical|center_horizontal"
    android:layout_alignBottom="@+id/NomCigala2"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:layout_marginRight="25dp"
    android:layout_marginEnd="25dp"
    android:layout_alignTop="@+id/NomCigala2" />
</RelativeLayout>

<RelativeLayout
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="90dp"
  android:layout_below="@+id/TerceraFila"
  android:layout_alignParentLeft="true"
  android:layout_alignParentStart="true"
  android:id="@+id/QuartaFila"
  android:background="@drawable/degradat_blanc">


  <ImageView
    android:id="@+id/CigalaCercle3"
    android:layout_width="65dp"
    android:layout_height="65dp"
    android:src="@drawable/hilaphura_varipes_cercle"
    android:layout_gravity="center"
    android:visibility="visible"
    android:layout_centerVertical="true"
    android:layout_marginLeft="20dp"/>

  <TextView
    android:layout_width="130dp"
    android:layout_height="40dp"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:text="@string/HilaphuraVaripes"
    android:id="@+id/NomCigala3"
    android:textColor="#010101"
    android:visibility="visible"
    android:gravity="left|center"
    android:textSize="18sp"
    android:layout_alignTop="@+id/CigalaCercle3"
    android:layout_toRightOf="@id/CigalaCercle3"
    android:layout_alignBottom="@+id/CigalaCercle3"
    android:layout_marginLeft="20dp"
    android:textStyle="italic" />

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="40dp"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:text="z%"
    android:id="@+id/PercentatgeSemblanca3"
    android:textColor="#010101"
    android:visibility="visible"
    android:textStyle="bold"
    android:textSize="24sp"
    android:gravity="center_vertical|center_horizontal"
    android:layout_alignBottom="@+id/NomCigala3"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:layout_marginRight="25dp"
    android:layout_marginEnd="25dp"
    android:layout_alignTop="@+id/NomCigala3" />
</RelativeLayout>

<RelativeLayout
  android:orientation="vertical"
  android:layout_width="match_parent"
  android:layout_height="90dp"
  android:layout_below="@+id/QuartaFila"
  android:layout_alignParentLeft="true"
  android:layout_alignParentStart="true"
  android:id="@+id/CinquenaFila"
  android:background="@drawable/degradat_blanc">


  <ImageView
    android:id="@+id/CigalaCercle4"
    android:layout_width="65dp"
    android:layout_height="65dp"
    android:src="@drawable/tibicina_haematodes_cercle"
    android:layout_gravity="center"
    android:visibility="visible"
    android:layout_centerVertical="true"
    android:layout_marginLeft="20dp"/>

  <TextView
    android:layout_width="130dp"
    android:layout_height="40dp"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:text="@string/TibicinaHaematodes"
    android:id="@+id/NomCigala4"
    android:textColor="#010101"
    android:visibility="visible"
    android:gravity="left|center"
    android:textSize="18sp"
    android:layout_alignTop="@+id/CigalaCercle4"
    android:layout_toRightOf="@id/CigalaCercle4"
    android:layout_alignBottom="@+id/CigalaCercle4"
    android:layout_marginLeft="20dp"
    android:textStyle="italic" />

  <TextView
    android:layout_width="wrap_content"
    android:layout_height="40dp"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:text="t%"
    android:id="@+id/PercentatgeSemblanca4"
    android:textColor="#010101"
    android:visibility="visible"
    android:textStyle="bold"
    android:textSize="24sp"
    android:gravity="center_vertical|center_horizontal"
    android:layout_alignBottom="@+id/NomCigala4"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:layout_marginRight="25dp"
    android:layout_marginEnd="25dp"
    android:layout_alignTop="@+id/NomCigala4" />
  </RelativeLayout>

  <RelativeLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="90dp"
    android:layout_below="@+id/CinquenaFila"
    android:layout_alignParentLeft="true"
    android:layout_alignParentStart="true"
    android:id="@+id/SisenaFila"
    android:background="#ffffff">


    <TextView
      android:layout_width="280dp"
      android:layout_height="90dp"
      android:textAppearance="?android:attr/textAppearanceSmall"
      android:text="@string/FrequenciaMostreigExigua"
      android:id="@+id/FrequenciaMostreigExigua"
      android:textColor="#010101"
      android:visibility="visible"
      android:textStyle="normal"
      android:textSize="10sp"
      android:gravity="center_vertical|left"
      android:layout_alignParentTop="true"
      android:layout_alignParentLeft="true"
      android:layout_alignParentStart="true"
      android:layout_marginLeft="23dp"
      android:layout_marginStart="23dp" />
  </RelativeLayout>

</RelativeLayout>
```

fitxa_cicada_barbara_lusitanica.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/CicBar_Titol"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/CicadaBarbaraLusitanica"
```

```
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#ffffff"
      android:id="@+id/CicBar_NomEspecie"
      android:layout_centerVertical="true"
      android:gravity="center_horizontal"
      android:layout_centerHorizontal="true"
      android:textSize="25sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="@dimen/activity_horizontal_margin"
   android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/CicBar_Imatge"
   android:layout_marginTop="0dp"
   android:layout_marginLeft="0dp"
   android:layout_marginRight="0dp"
   android:layout_below="@+id/CicBar_Titol"
   android:background="#ffffff">

   <ImageView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:adjustViewBounds="true"
      android:id="@+id/CicBar_ImatgeEspecie"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/cicada_barbara_lusitanica_370"
      android:layout_marginBottom="5dp"
      android:layout_marginTop="5dp"/>

   <TextView android:text="@string/PacoFaluke"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBar_AutoriaFotografia"
      android:layout_below="@+id/CicBar_ImatgeEspecie"
      android:layout_alignRight="@+id/CicBar_ImatgeEspecie"
      android:textSize="8sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/CicBar_Fenologia"
   android:layout_alignLeft="@+id/CicBar_Imatge"
   android:layout_marginTop="15dp"
   android:layout_marginLeft="20dp"
   android:layout_marginRight="20dp"
   android:layout_below="@+id/CicBar_Imatge"
   android:background="#c6c6c6">

   <TextView android:text="@string/Fenologia"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBar_TitolFenologia"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<ImageView
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:id="@+id/CicBar_ImatgeFenologia"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/fenologia_cicada_barbara_lusitanica"
   android:adjustViewBounds="true"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/CicBar_Fenologia"
   android:layout_alignLeft="@+id/CicBar_Fenologia"
   android:layout_alignRight="@+id/CicBar_Fenologia"
   android:layout_marginTop="10dp"
   android:layout_marginLeft="15dp"
   android:layout_marginRight="15dp"
   android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/CicBar_Identificacio"
   android:layout_alignLeft="@+id/CicBar_Fenologia"
   android:layout_alignRight="@+id/CicBar_Fenologia"
   android:layout_marginTop="15dp"
   android:layout_below="@+id/CicBar_ImatgeFenologia"
   android:background="#c6c6c6">

   <TextView android:text="@string/Identificacio"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBar_TitolIdentificacio"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioCicadaBarbaraLusitanica"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/CicBar_TextIdentificacio"
   android:layout_below="@+id/CicBar_Identificacio"
   android:layout_alignRight="@+id/CicBar_Identificacio"
   android:layout_alignLeft="@+id/CicBar_Identificacio"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/CicBar_Habitat"
   android:layout_alignLeft="@+id/CicBar_Identificacio"
   android:layout_alignRight="@+id/CicBar_Identificacio"
   android:layout_marginTop="17dp"
   android:layout_below="@+id/CicBar_TextIdentificacio"
   android:background="#c6c6c6">

   <TextView android:text="@string/Habitat"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBar_TitolHabitat"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/HabitatCicadaBarbaraLusitanica"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/CicBar_TextHabitat"
   android:layout_below="@+id/CicBar_Habitat"
   android:layout_alignLeft="@+id/CicBar_Habitat"
   android:layout_alignRight="@+id/CicBar_Habitat"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/CicBar_Cant"
   android:layout_alignLeft="@+id/CicBar_Habitat"
   android:layout_alignRight="@+id/CicBar_Habitat"
   android:layout_marginTop="17dp"
   android:layout_below="@+id/CicBar_TextHabitat"
   android:background="#c6c6c6">

   <TextView android:text="@string/Cant"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBar_TitolCant"
      android:textSize="15sp"
```

```
      android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/CantCicadaBarbaraLusitanica"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/CicBar_TextCant"
   android:layout_below="@+id/CicBar_Cant"
   android:layout_alignLeft="@+id/CicBar_Cant"
   android:layout_alignRight="@+id/CicBar_Cant"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>

<ImageView
   android:layout_width="match_parent"
   android:layout_height="45dp"
   android:id="@+id/CicBar_Cant1"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/cicada_barbara_lusitanica_cant1"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/CicBar_TextCant"
   android:layout_alignLeft="@+id/CicBar_Cant"
   android:layout_alignRight="@+id/CicBar_Cant"
   android:layout_marginTop="15dp"
   android:layout_marginLeft="15dp"
   android:layout_marginRight="15dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/CicBar_Cant1_Play"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/play_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/CicBar_Cant1"
   android:layout_marginTop="10dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/CicBar_Cant1_Pausa"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/pausa_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/CicBar_Cant1"
   android:layout_marginTop="10dp"
   android:textSize="12sp"
   android:visibility="gone"/>

<ImageView
   android:layout_width="match_parent"
   android:layout_height="45dp"
   android:id="@+id/CicBar_Cant2"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/cicada_barbara_lusitanica_cant2"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/CicBar_Cant1"
   android:layout_alignLeft="@+id/CicBar_Cant1"
   android:layout_alignRight="@+id/CicBar_Cant1"
   android:layout_marginTop="5dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/CicBar_Cant2_Play"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/play_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/CicBar_Cant2"
   android:layout_marginTop="10dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/CicBar_Cant2_Pausa"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/pausa_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/CicBar_Cant2"
   android:layout_marginTop="10dp"
   android:textSize="12sp"
   android:visibility="gone"/>

<TextView android:text="@string/FotografiaResultatsDanielRojas"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/CicBar_AutoriaFotografiaResultats"
   android:layout_below="@+id/CicBar_Cant2"
   android:layout_alignLeft="@+id/CicBar_Cant"
   android:layout_alignRight="@+id/CicBar_Cant"
   android:layout_marginTop="5dp"
   android:gravity="right"
   android:textSize="8sp"
   android:layout_marginLeft="0dp"/>


<TextView android:text=""
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/CicBar_EspaiBlanc"
   android:layout_below="@+id/CicBar_AutoriaFotografiaResultats"
   android:layout_alignLeft="@+id/CicBar_Cant2"
   android:layout_alignRight="@+id/CicBar_Cant2"
   android:layout_marginTop="5dp"
   android:textSize="12sp"
   android:layout_marginLeft="5dp"/>

</RelativeLayout>
</ScrollView>
```

fitxa_cicada_orni.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
   android:background="#ffffff">

   <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent" android:layout_height="match_parent">

      <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
         android:layout_width="match_parent"
         android:layout_height="60dp"
         android:paddingLeft="0dp"
         android:paddingRight="@dimen/activity_horizontal_margin"
         android:paddingTop="0dp"
         android:paddingBottom="0dp"
         android:id="@+id/CicOrn_Titol"
         android:layout_marginTop="0dp"
         android:layout_marginLeft="0dp"
         android:layout_marginRight="0dp"
         android:background="@drawable/degradat_blau">

         <TextView android:text="@string/CicadaOrni"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="#ffffff"
            android:id="@+id/CicOrn_NomEspecie"
            android:layout_centerVertical="true"
            android:gravity="center_horizontal"
            android:layout_centerHorizontal="true"
            android:textSize="26sp"/>
      </RelativeLayout>

      <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
         android:layout_width="match_parent"
         android:layout_height="match_parent"
         android:paddingLeft="@dimen/activity_horizontal_margin"
         android:paddingRight="@dimen/activity_horizontal_margin"
         android:paddingTop="0dp"
         android:paddingBottom="0dp"
         android:id="@+id/CicOrn_Imatge"
         android:layout_marginTop="0dp"
         android:layout_marginLeft="0dp"
         android:layout_marginRight="0dp"
         android:layout_below="@+id/CicOrn_Titol"
         android:background="#ffffff">

         <ImageView
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:id="@+id/CicOrn_ImatgeEspecie"
```

```
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/cicada_orni_370"
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_marginTop="5dp"/>

    <TextView android:text="@string/PerePons"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicOrn_AutoriaFotografia"
      android:layout_below="@+id/CicOrn_ImatgeEspecie"
      android:layout_alignRight="@+id/CicOrn_ImatgeEspecie"
      android:textSize="8sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicOrn_Fenologia"
    android:layout_alignLeft="@+id/CicOrn_Imatge"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_below="@+id/CicOrn_Imatge"
    android:background="#c6c6c6">

    <TextView android:text="@string/Fenologia"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicOrn_TitolFenologia"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/CicOrn_ImatgeFenologia"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/fenologia_cicada_orni"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicOrn_Fenologia"
    android:layout_alignLeft="@+id/CicOrn_Fenologia"
    android:layout_alignRight="@+id/CicOrn_Fenologia"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicOrn_Identificacio"
    android:layout_alignLeft="@+id/CicOrn_Fenologia"
    android:layout_alignRight="@+id/CicOrn_Fenologia"
    android:layout_marginTop="15dp"
    android:layout_below="@+id/CicOrn_ImatgeFenologia"
    android:background="#c6c6c6">

    <TextView android:text="@string/Identificacio"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicOrn_TitolIdentificacio"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioCicadaOrni"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicOrn_TextIdentificacio"
    android:layout_below="@+id/CicOrn_Identificacio"
    android:layout_alignLeft="@+id/CicOrn_Identificacio"
    android:layout_alignRight="@+id/CicOrn_Identificacio"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="120dp"
    android:layout_height="match_parent"
    android:id="@+id/CicOrn_ImatgeIdentificacio"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/identificacio_cicada_orni"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicOrn_TextIdentificacio"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<TextView android:text="@string/FerranTurmo"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicOrn_AutoriaFotografiaIdentificacio"
    android:layout_below="@+id/CicOrn_ImatgeIdentificacio"
    android:layout_alignLeft="@+id/CicOrn_ImatgeIdentificacio"
    android:layout_alignRight="@+id/CicOrn_ImatgeIdentificacio"
    android:gravity="right"
    android:textSize="7sp"
    android:layout_marginLeft="0dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicOrn_Habitat"
    android:layout_alignLeft="@+id/CicOrn_Identificacio"
    android:layout_alignRight="@+id/CicOrn_Identificacio"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/CicOrn_AutoriaFotografiaIdentificacio"
    android:background="#c6c6c6">

    <TextView android:text="@string/Habitat"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicOrn_TitolHabitat"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/HabitatCicadaOrni"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicOrn_TextHabitat"
    android:layout_below="@+id/CicOrn_Habitat"
    android:layout_alignLeft="@+id/CicOrn_Habitat"
    android:layout_alignRight="@+id/CicOrn_Habitat"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicOrn_Cant"
    android:layout_alignLeft="@+id/CicOrn_Habitat"
    android:layout_alignRight="@+id/CicOrn_Habitat"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/CicOrn_TextHabitat"
    android:background="#c6c6c6">

    <TextView android:text="@string/Cant"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicOrn_TitolCant"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
```

```
</RelativeLayout>


<TextView android:text="@string/CantCicadaOrni"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/CicOrn_TextCant"
  android:layout_below="@+id/CicOrn_Cant"
  android:layout_alignLeft="@+id/CicOrn_Cant"
  android:layout_alignRight="@+id/CicOrn_Cant"
  android:layout_marginTop="8dp"
  android:textSize="13sp"
  android:layout_marginLeft="5dp"/>

<ImageView
  android:layout_width="match_parent"
  android:layout_height="45dp"
  android:id="@+id/CicOrn_Cant1"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/cicada_orni_cant1"
  android:layout_marginBottom="5dp"
  android:layout_below="@+id/CicOrn_TextCant"
  android:layout_alignLeft="@+id/CicOrn_Cant"
  android:layout_alignRight="@+id/CicOrn_Cant"
  android:layout_marginTop="15dp"
  android:layout_marginLeft="15dp"
  android:layout_marginRight="15dp"
  android:textSize="12sp"/>

<ImageView
  android:layout_width="25dp"
  android:layout_height="25dp"
  android:id="@+id/CicOrn_Cant1_Play"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/play_gris"
  android:layout_marginBottom="5dp"
  android:layout_alignTop="@+id/CicOrn_Cant1"
  android:layout_marginTop="10dp"
  android:textSize="12sp"/>

<ImageView
  android:layout_width="25dp"
  android:layout_height="25dp"
  android:id="@+id/CicOrn_Cant1_Pausa"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/pausa_gris"
  android:layout_marginBottom="5dp"
  android:layout_alignTop="@+id/CicOrn_Cant1"
  android:layout_marginTop="10dp"
  android:textSize="12sp"
  android:visibility="gone"/>

<ImageView
  android:layout_width="match_parent"
  android:layout_height="45dp"
  android:id="@+id/CicOrn_Cant2"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/cicada_orni_cant2"
  android:layout_marginBottom="5dp"
  android:layout_below="@+id/CicOrn_Cant1"
  android:layout_alignLeft="@+id/CicOrn_Cant1"
  android:layout_alignRight="@+id/CicOrn_Cant1"
  android:layout_marginTop="5dp"
  android:textSize="12sp"/>

<ImageView
  android:layout_width="25dp"
  android:layout_height="25dp"
  android:id="@+id/CicOrn_Cant2_Play"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/play_gris"
  android:layout_marginBottom="5dp"
  android:layout_alignTop="@+id/CicOrn_Cant2"
  android:layout_marginTop="10dp"
  android:textSize="12sp"/>

<ImageView
  android:layout_width="25dp"
  android:layout_height="25dp"
  android:id="@+id/CicOrn_Cant2_Pausa"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/pausa_gris"
  android:layout_marginBottom="5dp"
  android:layout_alignTop="@+id/CicOrn_Cant2"
  android:layout_marginTop="10dp"
  android:textSize="12sp"
  android:visibility="gone"/>


  <TextView android:text=""
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/CicOrn_EspaiBlanc"
  android:layout_below="@+id/CicOrn_Cant2"
  android:layout_alignLeft="@+id/CicOrn_Cant2"
  android:layout_alignRight="@+id/CicOrn_Cant2"
  android:layout_marginTop="5dp"
  android:textSize="12sp"
  android:layout_marginLeft="5dp"/>
 </RelativeLayout>
</ScrollView>
```

fitxa_cicadatra_atra.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/CicAtr_Titol"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/CicadatraAtra"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/CicAtr_NomEspecie"
        android:layout_centerVertical="true"
        android:gravity="center_horizontal"
        android:layout_centerHorizontal="true"
        android:textSize="26sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/CicAtr_Imatge"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:layout_below="@+id/CicAtr_Titol"
      android:background="#ffffff">

      <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/CicAtr_ImatgeEspecie"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/cicadatra_atra_370"
        android:adjustViewBounds="true"
        android:layout_marginBottom="5dp"
        android:layout_marginTop="5dp"/>

      <TextView android:text="@string/XavierDeYzaguirre"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/CicAtr_AutoriaFotografia"
        android:layout_below="@+id/CicAtr_ImatgeEspecie"
        android:layout_alignRight="@+id/CicAtr_ImatgeEspecie"
        android:textSize="8sp"/>
```

```
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/CicAtr_Fenologia"
  android:layout_alignLeft="@+id/CicAtr_Imatge"
  android:layout_marginTop="15dp"
  android:layout_marginLeft="20dp"
  android:layout_marginRight="20dp"
  android:layout_below="@+id/CicAtr_Imatge"
  android:background="#c6c6c6">

  <TextView android:text="@string/Fenologia"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicAtr_TitolFenologia"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<ImageView
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:id="@+id/CicAtr_ImatgeFenologia"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/fenologia_cicadatra_atra"
  android:adjustViewBounds="true"
  android:layout_marginBottom="5dp"
  android:layout_below="@+id/CicAtr_Fenologia"
  android:layout_alignLeft="@+id/CicAtr_Fenologia"
  android:layout_alignRight="@+id/CicAtr_Fenologia"
  android:layout_marginTop="10dp"
  android:layout_marginLeft="15dp"
  android:layout_marginRight="15dp"
  android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/CicAtr_Identificacio"
  android:layout_alignLeft="@+id/CicAtr_Fenologia"
  android:layout_alignRight="@+id/CicAtr_Fenologia"
  android:layout_marginTop="15dp"
  android:layout_below="@+id/CicAtr_ImatgeFenologia"
  android:background="#c6c6c6">

  <TextView android:text="@string/Identificacio"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicAtr_TitolIdentificacio"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioCicadatraAtra"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/CicAtr_TextIdentificacio"
  android:layout_below="@+id/CicAtr_Identificacio"
  android:layout_alignLeft="@+id/CicAtr_Identificacio"
  android:layout_alignRight="@+id/CicAtr_Identificacio"
  android:layout_marginTop="8dp"
  android:textSize="13sp"
  android:layout_marginLeft="5dp"/>

<ImageView
  android:layout_width="120dp"
  android:layout_height="match_parent"
  android:id="@+id/CicAtr_ImatgeIdentificacio"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/identificacio_cicadatra_atra"
  android:adjustViewBounds="true"
  android:layout_marginBottom="5dp"
  android:layout_below="@+id/CicAtr_TextIdentificacio"
  android:layout_marginTop="10dp"
  android:layout_marginLeft="5dp"
  android:layout_marginRight="15dp"
  android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/CicAtr_Habitat"
  android:layout_alignLeft="@+id/CicAtr_Identificacio"
  android:layout_alignRight="@+id/CicAtr_Identificacio"
  android:layout_marginTop="17dp"
  android:layout_below="@+id/CicAtr_ImatgeIdentificacio"
  android:background="#c6c6c6">

  <TextView android:text="@string/Habitat"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicAtr_TitolHabitat"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/HabitatCicadatraAtra"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/CicAtr_TextHabitat"
  android:layout_below="@+id/CicAtr_Habitat"
  android:layout_alignLeft="@+id/CicAtr_Habitat"
  android:layout_alignRight="@+id/CicAtr_Habitat"
  android:layout_marginTop="8dp"
  android:textSize="13sp"
  android:layout_marginLeft="5dp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/CicAtr_Cant"
  android:layout_alignLeft="@+id/CicAtr_Habitat"
  android:layout_alignRight="@+id/CicAtr_Habitat"
  android:layout_marginTop="17dp"
  android:layout_below="@+id/CicAtr_TextHabitat"
  android:background="#c6c6c6">

  <TextView android:text="@string/Cant"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicAtr_TitolCant"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/CantCicadatraAtra"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/CicAtr_TextCant"
  android:layout_below="@+id/CicAtr_Cant"
  android:layout_alignLeft="@+id/CicAtr_Cant"
  android:layout_alignRight="@+id/CicAtr_Cant"
  android:layout_marginTop="8dp"
  android:textSize="13sp"
  android:layout_marginLeft="5dp"/>

<ImageView
  android:layout_width="match_parent"
  android:layout_height="45dp"
  android:id="@+id/CicAtr_Cant1"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/cicadatra_atra_cant1"
  android:layout_marginBottom="5dp"
  android:layout_below="@+id/CicAtr_TextCant"
  android:layout_alignLeft="@+id/CicAtr_Cant"
  android:layout_alignRight="@+id/CicAtr_Cant"
```

```
    android:layout_marginTop="15dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicAtr_Cant1_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicAtr_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicAtr_Cant1_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicAtr_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/CicAtr_Cant2"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/cicadatra_atra_cant2"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicAtr_Cant1"
    android:layout_alignLeft="@+id/CicAtr_Cant1"
    android:layout_alignRight="@+id/CicAtr_Cant1"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicAtr_Cant2_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicAtr_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicAtr_Cant2_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicAtr_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/CicAtr_Cant3"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/cicadatra_atra_cant3"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicAtr_Cant2"
    android:layout_alignLeft="@+id/CicAtr_Cant2"
    android:layout_alignRight="@+id/CicAtr_Cant2"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicAtr_Cant3_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicAtr_Cant3"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicAtr_Cant3_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicAtr_Cant3"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>


<TextView android:text=""
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicAtr_EspaiBlanc"
    android:layout_below="@+id/CicAtr_Cant3"
    android:layout_alignLeft="@+id/CicAtr_Cant3"
    android:layout_alignRight="@+id/CicAtr_Cant3"
    android:layout_marginTop="5dp"
    android:textSize="12sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>
</ScrollView>
```

fitxa_cicadetta_brevipennis.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#ffffff">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent" android:layout_height="match_parent">

        <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="match_parent"
            android:layout_height="60dp"
            android:paddingLeft="0dp"
            android:paddingRight="@dimen/activity_horizontal_margin"
            android:paddingTop="0dp"
            android:paddingBottom="0dp"
            android:id="@+id/CicBre_Titol"
            android:layout_marginTop="0dp"
            android:layout_marginLeft="0dp"
            android:layout_marginRight="0dp"
            android:background="@drawable/degradat_blau">

            <TextView android:text="@string/CicadettaBrevipennis"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textColor="#ffffff"
                android:id="@+id/CicBre_NomEspecie"
                android:layout_centerVertical="true"
                android:gravity="center_horizontal"
                android:layout_centerHorizontal="true"
                android:textSize="26sp"/>
        </RelativeLayout>

        <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingLeft="@dimen/activity_horizontal_margin"
            android:paddingRight="@dimen/activity_horizontal_margin"
            android:paddingTop="0dp"
            android:paddingBottom="0dp"
            android:id="@+id/CicBre_Imatge"
            android:layout_marginTop="0dp"
            android:layout_marginLeft="0dp"
            android:layout_marginRight="0dp"
            android:layout_below="@+id/CicBre_Titol"
            android:background="#ffffff">

            <ImageView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/CicBre_ImatgeEspecie"
                android:layout_centerHorizontal="true"
                android:layout_centerVertical="false"
                android:src="@drawable/cicadetta_brevipennis_370"
```

```
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_marginTop="5dp"/>

    <TextView android:text="@string/MatijaGogala"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBre_AutoriaFotografia"
      android:layout_below="@+id/CicBre_ImatgeEspecie"
      android:layout_alignRight="@+id/CicBre_ImatgeEspecie"
      android:textSize="8sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicBre_Fenologia"
    android:layout_alignLeft="@+id/CicBre_Imatge"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_below="@+id/CicBre_Imatge"
    android:background="#c6c6c6">

    <TextView android:text="@string/Fenologia"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBre_TitolFenologia"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/CicBre_ImatgeFenologia"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/fenologia_cicadetta_brevipennis"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicBre_Fenologia"
    android:layout_alignLeft="@+id/CicBre_Fenologia"
    android:layout_alignRight="@+id/CicBre_Fenologia"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicBre_Identificacio"
    android:layout_alignLeft="@+id/CicBre_Fenologia"
    android:layout_alignRight="@+id/CicBre_Fenologia"
    android:layout_marginTop="15dp"
    android:layout_below="@+id/CicBre_ImatgeFenologia"
    android:background="#c6c6c6">

    <TextView android:text="@string/Identificacio"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBre_TitolIdentificacio"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioCicadettaBrevipennis"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicBre_TextIdentificacio"
    android:layout_below="@+id/CicBre_Identificacio"
    android:layout_alignLeft="@+id/CicBre_Identificacio"
    android:layout_alignRight="@+id/CicBre_Identificacio"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicBre_Habitat"
    android:layout_alignLeft="@+id/CicBre_Identificacio"
    android:layout_alignRight="@+id/CicBre_Identificacio"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/CicBre_TextIdentificacio"
    android:background="#c6c6c6">

    <TextView android:text="@string/Habitat"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBre_TitolHabitat"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/HabitatCicadettaBrevipennis"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicBre_TextHabitat"
    android:layout_below="@+id/CicBre_Habitat"
    android:layout_alignLeft="@+id/CicBre_Habitat"
    android:layout_alignRight="@+id/CicBre_Habitat"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicBre_Cant"
    android:layout_alignLeft="@+id/CicBre_Habitat"
    android:layout_alignRight="@+id/CicBre_Habitat"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/CicBre_TextHabitat"
    android:background="#c6c6c6">

    <TextView android:text="@string/Cant"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBre_TitolCant"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/CantCicadettaBrevipennis"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicBre_TextCant"
    android:layout_below="@+id/CicBre_Cant"
    android:layout_alignLeft="@+id/CicBre_Cant"
    android:layout_alignRight="@+id/CicBre_Cant"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/CicBre_Cant1"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/cicadetta_brevipennis_cant1"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicBre_TextCant"
    android:layout_alignLeft="@+id/CicBre_Cant"
    android:layout_alignRight="@+id/CicBre_Cant"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
```

```
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicBre_Cant1_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicBre_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicBre_Cant1_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicBre_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/CicBre_Cant2"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/cicadetta_brevipennis_cant2"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicBre_Cant1"
    android:layout_alignLeft="@+id/CicBre_Cant1"
    android:layout_alignRight="@+id/CicBre_Cant1"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicBre_Cant2_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicBre_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicBre_Cant2_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicBre_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/CicBre_Cant3"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/cicadetta_brevipennis_cant3"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicBre_Cant2"
    android:layout_alignLeft="@+id/CicBre_Cant2"
    android:layout_alignRight="@+id/CicBre_Cant2"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/CicBre_Cant3_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/CicBre_Cant3"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

    <ImageView
      android:layout_width="25dp"
      android:layout_height="25dp"
      android:id="@+id/CicBre_Cant3_Pausa"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/pausa_gris"
      android:layout_marginBottom="5dp"
      android:layout_alignTop="@+id/CicBre_Cant3"
      android:layout_marginTop="10dp"
      android:textSize="12sp"
      android:visibility="gone"/>


    <TextView android:text=""
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/CicBre_EspaiBlanc"
      android:layout_below="@+id/CicBre_Cant3"
      android:layout_alignLeft="@+id/CicBre_Cant3"
      android:layout_alignRight="@+id/CicBre_Cant3"
      android:layout_marginTop="5dp"
      android:textSize="12sp"
      android:layout_marginLeft="5dp"/>
  </RelativeLayout>
</ScrollView>
```

fitxa_cicadetta_cerdaniensis.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/CicCer_Titol"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/CicadettaCerdaniensis"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/CicCer_NomEspecie"
        android:layout_centerVertical="true"
        android:gravity="center_horizontal"
        android:layout_centerHorizontal="true"
        android:textSize="26sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/CicCer_Imatge"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:layout_below="@+id/CicCer_Titol"
      android:background="#ffffff">

      <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/CicCer_ImatgeEspecie"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/cicadetta_cerdaniensis_370"
        android:adjustViewBounds="true"
        android:layout_marginBottom="5dp"
        android:layout_marginTop="5dp"/>
```

```
    <TextView android:text="@string/ThomasHertach"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/CicCer_AutoriaFotografia"
        android:layout_below="@+id/CicCer_ImatgeEspecie"
        android:layout_alignRight="@+id/CicCer_ImatgeEspecie"
        android:textSize="8sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicCer_Fenologia"
    android:layout_alignLeft="@+id/CicCer_Imatge"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_below="@+id/CicCer_Imatge"
    android:background="#c6c6c6">

    <TextView android:text="@string/Fenologia"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/CicCer_TitolFenologia"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/CicCer_ImatgeFenologia"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/fenologia_cicadetta_cerdaniensis"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicCer_Fenologia"
    android:layout_alignLeft="@+id/CicCer_Fenologia"
    android:layout_alignRight="@+id/CicCer_Fenologia"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicCer_Identificacio"
    android:layout_alignLeft="@+id/CicCer_Fenologia"
    android:layout_alignRight="@+id/CicCer_Fenologia"
    android:layout_marginTop="15dp"
    android:layout_below="@+id/CicCer_ImatgeFenologia"
    android:background="#c6c6c6">

    <TextView android:text="@string/Identificacio"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/CicCer_TitolIdentificacio"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioCicadettaCerdaniensis"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicCer_TextIdentificacio"
    android:layout_below="@+id/CicCer_Identificacio"
    android:layout_alignLeft="@+id/CicCer_Identificacio"
    android:layout_alignRight="@+id/CicCer_Identificacio"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicCer_Habitat"
    android:layout_alignLeft="@+id/CicCer_Identificacio"
    android:layout_alignRight="@+id/CicCer_Identificacio"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/CicCer_TextIdentificacio"
    android:background="#c6c6c6">

    <TextView android:text="@string/Habitat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/CicCer_TitolHabitat"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/HabitatCicadettaCerdaniensis"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicCer_TextHabitat"
    android:layout_below="@+id/CicCer_Habitat"
    android:layout_alignLeft="@+id/CicCer_Habitat"
    android:layout_alignRight="@+id/CicCer_Habitat"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/CicCer_Cant"
    android:layout_alignLeft="@+id/CicCer_Habitat"
    android:layout_alignRight="@+id/CicCer_Habitat"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/CicCer_TextHabitat"
    android:background="#c6c6c6">

    <TextView android:text="@string/Cant"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/CicCer_TitolCant"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/CantCicadettaCerdaniensis"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/CicCer_TextCant"
    android:layout_below="@+id/CicCer_Cant"
    android:layout_alignLeft="@+id/CicCer_Cant"
    android:layout_alignRight="@+id/CicCer_Cant"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/CicCer_Cant1"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/cicadetta_cerdaniensis_cant1"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/CicCer_TextCant"
    android:layout_alignLeft="@+id/CicCer_Cant"
    android:layout_alignRight="@+id/CicCer_Cant"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<ImageView
```

```
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/CicCer_Cant1_Play"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/play_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/CicCer_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/CicCer_Cant1_Pausa"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/pausa_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/CicCer_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"
        android:visibility="gone"/>

    <TextView android:text=""
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/CicCer_EspaiBlanc"
        android:layout_below="@+id/CicCer_Cant1"
        android:layout_alignLeft="@+id/CicCer_Cant1"
        android:layout_alignRight="@+id/CicCer_Cant1"
        android:layout_marginTop="5dp"
        android:textSize="12sp"
        android:layout_marginLeft="5dp"/>
  </RelativeLayout>
</ScrollView>
```

fitxa_hilaphura_varipes.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/HilVar_Titol"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/HilaphuraVaripes"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/HilVar_NomEspecie"
        android:layout_centerVertical="true"
        android:gravity="center_horizontal"
        android:layout_centerHorizontal="true"
        android:textSize="26sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/HilVar_Imatge"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:layout_below="@+id/HilVar_Titol"
      android:background="#ffffff">

      <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/HilVar_ImatgeEspecie"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/hilaphura_varipes_370"
        android:adjustViewBounds="true"
        android:layout_marginBottom="5dp"
        android:layout_marginTop="5dp"/>

      <TextView android:text="@string/AntonioGarciaMaldonado"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/HilVar_AutoriaFotografia"
        android:layout_below="@+id/HilVar_ImatgeEspecie"
        android:layout_alignRight="@+id/HilVar_ImatgeEspecie"
        android:textSize="8sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/HilVar_Fenologia"
      android:layout_alignLeft="@+id/HilVar_Imatge"
      android:layout_marginTop="15dp"
      android:layout_marginLeft="20dp"
      android:layout_marginRight="20dp"
      android:layout_below="@+id/HilVar_Imatge"
      android:background="#c6c6c6">

      <TextView android:text="@string/Fenologia"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/HilVar_TitolFenologia"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>

    <ImageView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/HilVar_ImatgeFenologia"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/fenologia_hilaphura_varipes"
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_below="@+id/HilVar_Fenologia"
      android:layout_alignLeft="@+id/HilVar_Fenologia"
      android:layout_alignRight="@+id/HilVar_Fenologia"
      android:layout_marginTop="10dp"
      android:layout_marginLeft="15dp"
      android:layout_marginRight="15dp"
      android:textSize="12sp"/>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/HilVar_Identificacio"
      android:layout_alignLeft="@+id/HilVar_Fenologia"
      android:layout_alignRight="@+id/HilVar_Fenologia"
      android:layout_marginTop="15dp"
      android:layout_below="@+id/HilVar_ImatgeFenologia"
      android:background="#c6c6c6">

      <TextView android:text="@string/Identificacio"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/HilVar_TitolIdentificacio"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>

    <TextView android:text="@string/IdentificacioHilaphuraVaripes"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/HilVar_TextIdentificacio"
      android:layout_below="@+id/HilVar_Identificacio"
      android:layout_alignLeft="@+id/HilVar_Identificacio"
```

```
    android:layout_alignRight="@+id/HilVar_Identificacio"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="120dp"
    android:layout_height="match_parent"
    android:id="@+id/HilVar_ImatgeIdentificacio"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/identificacio_hilaphura_varipes"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/HilVar_TextIdentificacio"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<TextView android:text="@string/PerePons"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/HilVar_AutoriaFotografiaIdentificacio"
    android:layout_below="@+id/HilVar_ImatgeIdentificacio"
    android:layout_alignLeft="@+id/HilVar_ImatgeIdentificacio"
    android:layout_alignRight="@+id/HilVar_ImatgeIdentificacio"
    android:gravity="right"
    android:textSize="7sp"
    android:layout_marginLeft="0dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/HilVar_Habitat"
    android:layout_alignLeft="@+id/HilVar_Identificacio"
    android:layout_alignRight="@+id/HilVar_Identificacio"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/HilVar_AutoriaFotografiaIdentificacio"
    android:background="#c6c6c6">

    <TextView android:text="@string/Habitat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/HilVar_TitolHabitat"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/HabitatHilaphuraVaripes"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/HilVar_TextHabitat"
    android:layout_below="@+id/HilVar_Habitat"
    android:layout_alignLeft="@+id/HilVar_Habitat"
    android:layout_alignRight="@+id/HilVar_Habitat"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/HilVar_Cant"
    android:layout_alignLeft="@+id/HilVar_Habitat"
    android:layout_alignRight="@+id/HilVar_Habitat"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/HilVar_TextHabitat"
    android:background="#c6c6c6">

    <TextView android:text="@string/Cant"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/HilVar_TitolCant"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



    <TextView android:text="@string/CantHilaphuraVaripes"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/HilVar_TextCant"
        android:layout_below="@+id/HilVar_Cant"
        android:layout_alignLeft="@+id/HilVar_Cant"
        android:layout_alignRight="@+id/HilVar_Cant"
        android:layout_marginTop="8dp"
        android:textSize="13sp"
        android:layout_marginLeft="5dp"/>

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:id="@+id/HilVar_Cant1"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/hilaphura_varipes_cant1"
        android:layout_marginBottom="5dp"
        android:layout_below="@+id/HilVar_TextCant"
        android:layout_alignLeft="@+id/HilVar_Cant"
        android:layout_alignRight="@+id/HilVar_Cant"
        android:layout_marginTop="15dp"
        android:layout_marginLeft="15dp"
        android:layout_marginRight="15dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/HilVar_Cant1_Play"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/play_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/HilVar_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/HilVar_Cant1_Pausa"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/pausa_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/HilVar_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"
        android:visibility="gone"/>

    <TextView android:text="@string/FotografiaResultatsIvanJesusTorresanoGarcia"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/HilVar_AutoriaFotografiaResultats"
        android:layout_below="@+id/HilVar_Cant1"
        android:layout_alignLeft="@+id/HilVar_Cant1"
        android:layout_alignRight="@+id/HilVar_Cant1"
        android:layout_marginTop="5dp"
        android:gravity="right"
        android:textSize="8sp"
        android:layout_marginLeft="0dp"/>

    <TextView android:text=""
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/HilVar_EspaiBlanc"
        android:layout_below="@+id/HilVar_AutoriaFotografiaResultats"
        android:layout_alignLeft="@+id/HilVar_Cant1"
        android:layout_alignRight="@+id/HilVar_Cant1"
        android:layout_marginTop="5dp"
        android:textSize="12sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>
</ScrollView>
```

fitxa_lyristes_plebejus.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/LyrPle_Titol"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/LyristesPlebejus"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/LyrPle_NomEspecie"
        android:layout_centerVertical="true"
        android:gravity="center_horizontal"
        android:layout_centerHorizontal="true"
        android:textSize="26sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/LyrPle_Imatge"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:layout_below="@+id/LyrPle_Titol"
      android:background="#ffffff">

      <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/LyrPle_ImatgeEspecie"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/lyristes_plebejus_370"
        android:adjustViewBounds="true"
        android:layout_marginBottom="5dp"
        android:layout_marginTop="5dp"/>

      <TextView android:text="@string/PerePons"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/LyrPle_AutoriaFotografia"
        android:layout_below="@+id/LyrPle_ImatgeEspecie"
        android:layout_alignRight="@+id/LyrPle_ImatgeEspecie"
        android:textSize="8sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/LyrPle_Fenologia"
      android:layout_alignLeft="@+id/LyrPle_Imatge"
      android:layout_marginTop="15dp"
      android:layout_marginLeft="20dp"
      android:layout_marginRight="20dp"
      android:layout_below="@+id/LyrPle_Imatge"
      android:background="#c6c6c6">

      <TextView android:text="@string/Fenologia"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/LyrPle_TitolFenologia"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>

    <ImageView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/LyrPle_ImatgeFenologia"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/fenologia_lyristes_plebejus"
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_below="@+id/LyrPle_Fenologia"
      android:layout_alignLeft="@+id/LyrPle_Fenologia"
      android:layout_alignRight="@+id/LyrPle_Fenologia"
      android:layout_marginTop="10dp"
      android:layout_marginLeft="15dp"
      android:layout_marginRight="15dp"
      android:textSize="12sp"/>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/LyrPle_Identificacio"
      android:layout_alignLeft="@+id/LyrPle_Fenologia"
      android:layout_alignRight="@+id/LyrPle_Fenologia"
      android:layout_marginTop="15dp"
      android:layout_below="@+id/LyrPle_ImatgeFenologia"
      android:background="#c6c6c6">

      <TextView android:text="@string/Identificacio"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/LyrPle_TitolIdentificacio"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>

    <TextView android:text="@string/IdentificacioLyristesPlebejus"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/LyrPle_TextIdentificacio"
      android:layout_below="@+id/LyrPle_Identificacio"
      android:layout_alignLeft="@+id/LyrPle_Identificacio"
      android:layout_alignRight="@+id/LyrPle_Identificacio"
      android:layout_marginTop="8dp"
      android:textSize="13sp"
      android:layout_marginLeft="5dp"/>

    <ImageView
      android:layout_width="150dp"
      android:layout_height="match_parent"
      android:id="@+id/LyrPle_ImatgeIdentificacio"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/identificacio_lyristes_plebejus"
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_below="@+id/LyrPle_TextIdentificacio"
      android:layout_marginTop="10dp"
      android:layout_marginLeft="5dp"
      android:layout_marginRight="15dp"
      android:textSize="12sp"/>

    <TextView android:text="@string/JeanPierreLavigne"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/LyrPle_AutoriaFotografiaIdentificacio"
      android:layout_below="@+id/LyrPle_ImatgeIdentificacio"
      android:layout_alignLeft="@+id/LyrPle_ImatgeIdentificacio"
      android:layout_alignRight="@+id/LyrPle_ImatgeIdentificacio"
      android:gravity="right"
      android:textSize="7sp"
      android:layout_marginLeft="0dp"/>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/LyrPle_Habitat"
      android:layout_alignLeft="@+id/LyrPle_Identificacio"
```

```
    android:layout_alignRight="@+id/LyrPle_Identificacio"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/LyrPle_AutoriaFotografiaIdentificacio"
    android:background="#c6c6c6">

    <TextView android:text="@string/Habitat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/LyrPle_TitolHabitat"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/HabitatLyristesPlebejus"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/LyrPle_TextHabitat"
    android:layout_below="@+id/LyrPle_Habitat"
    android:layout_alignLeft="@+id/LyrPle_Habitat"
    android:layout_alignRight="@+id/LyrPle_Habitat"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/LyrPle_Cant"
    android:layout_alignLeft="@+id/LyrPle_Habitat"
    android:layout_alignRight="@+id/LyrPle_Habitat"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/LyrPle_TextHabitat"
    android:background="#c6c6c6">

    <TextView android:text="@string/Cant"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/LyrPle_TitolCant"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/CantLyristesPlebejus"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/LyrPle_TextCant"
    android:layout_below="@+id/LyrPle_Cant"
    android:layout_alignLeft="@+id/LyrPle_Cant"
    android:layout_alignRight="@+id/LyrPle_Cant"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/LyrPle_Cant1"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/lyristes_plebejus_cant1"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/LyrPle_TextCant"
    android:layout_alignLeft="@+id/LyrPle_Cant"
    android:layout_alignRight="@+id/LyrPle_Cant"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/LyrPle_Cant1_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/LyrPle_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/LyrPle_Cant1_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/LyrPle_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/LyrPle_Cant2"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/lyristes_plebejus_cant2"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/LyrPle_Cant1"
    android:layout_alignLeft="@+id/LyrPle_Cant1"
    android:layout_alignRight="@+id/LyrPle_Cant1"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/LyrPle_Cant2_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/LyrPle_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/LyrPle_Cant2_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/LyrPle_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<TextView android:text="@string/FotografiaResultatsCosminOvidiu"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/LyrPle_AutoriaFotografiaResultats"
    android:layout_below="@+id/LyrPle_Cant2"
    android:layout_alignLeft="@+id/LyrPle_Cant"
    android:layout_alignRight="@+id/LyrPle_Cant"
    android:layout_marginTop="5dp"
    android:gravity="right"
    android:textSize="8sp"
    android:layout_marginLeft="0dp"/>

<TextView android:text=""
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/LyrPle_EspaiBlanc"
    android:layout_below="@+id/LyrPle_AutoriaFotografiaResultats"
    android:layout_alignLeft="@+id/LyrPle_Cant2"
    android:layout_alignRight="@+id/LyrPle_Cant2"
    android:layout_marginTop="5dp"
    android:textSize="12sp"
    android:layout_marginLeft="5dp"/>
  </RelativeLayout>
</ScrollView>
```

fitxa_tettigettalna_argentata.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
```

```
android:layout_height="fill_parent"
android:background="#ffffff">

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent" android:layout_height="match_parent">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetArg_Titol"
    android:layout_marginTop="0dp"
    android:layout_marginLeft="0dp"
    android:layout_marginRight="0dp"
    android:background="@drawable/degradat_blau">

    <TextView android:text="@string/TettigettalnaArgentata"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#ffffff"
      android:id="@+id/TetArg_NomEspecie"
      android:layout_centerVertical="true"
      android:gravity="center_horizontal"
      android:layout_centerHorizontal="true"
      android:textSize="26sp"/>
  </RelativeLayout>

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetArg_Imatge"
    android:layout_marginTop="0dp"
    android:layout_marginLeft="0dp"
    android:layout_marginRight="0dp"
    android:layout_below="@+id/TetArg_Titol"
    android:background="#ffffff">

    <ImageView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/TetArg_ImatgeEspecie"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/tettigettalna_argentata_370"
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_marginTop="5dp"/>

    <TextView android:text="@string/PerePons"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TetArg_AutoriaFotografia"
      android:layout_below="@+id/TetArg_ImatgeEspecie"
      android:layout_alignRight="@+id/TetArg_ImatgeEspecie"
      android:textSize="8sp"/>
  </RelativeLayout>

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetArg_Fenologia"
    android:layout_alignLeft="@+id/TetArg_Imatge"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_below="@+id/TetArg_Imatge"
    android:background="#c6c6c6">

    <TextView android:text="@string/Fenologia"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TetArg_TitolFenologia"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
  </RelativeLayout>

  <ImageView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/TetArg_ImatgeFenologia"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/fenologia_tettigettalna_argentata"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TetArg_Fenologia"
    android:layout_alignLeft="@+id/TetArg_Fenologia"
    android:layout_alignRight="@+id/TetArg_Fenologia"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetArg_Identificacio"
    android:layout_alignLeft="@+id/TetArg_Fenologia"
    android:layout_alignRight="@+id/TetArg_Fenologia"
    android:layout_marginTop="15dp"
    android:layout_below="@+id/TetArg_ImatgeFenologia"
    android:background="#c6c6c6">

    <TextView android:text="@string/Identificacio"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TetArg_TitolIdentificacio"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
  </RelativeLayout>

  <TextView android:text="@string/IdentificacioTettigettalnaArgentata"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TetArg_TextIdentificacio"
    android:layout_below="@+id/TetArg_Identificacio"
    android:layout_alignLeft="@+id/TetArg_Identificacio"
    android:layout_alignRight="@+id/TetArg_Identificacio"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

  <ImageView
    android:layout_width="120dp"
    android:layout_height="match_parent"
    android:id="@+id/TetArg_ImatgeIdentificacio"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/identificacio_tettigettalna_argentata"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TetArg_TextIdentificacio"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

  <TextView android:text="@string/JoseManuelSesma"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TetArg_AutoriaFotografiaIdentificacio"
    android:layout_below="@+id/TetArg_ImatgeIdentificacio"
    android:layout_alignLeft="@+id/TetArg_ImatgeIdentificacio"
    android:layout_alignRight="@+id/TetArg_ImatgeIdentificacio"
    android:gravity="right"
    android:textSize="7sp"
    android:layout_marginLeft="0dp"/>



  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetArg_Habitat"
    android:layout_alignLeft="@+id/TetArg_Identificacio"
    android:layout_alignRight="@+id/TetArg_Identificacio"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/TetArg_AutoriaFotografiaIdentificacio"
```

```
    android:background="#c6c6c6">

    <TextView android:text="@string/Habitat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetArg_TitolHabitat"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/HabitatTettigettalnaArgentata"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TetArg_TextHabitat"
    android:layout_below="@+id/TetArg_Habitat"
    android:layout_alignLeft="@+id/TetArg_Habitat"
    android:layout_alignRight="@+id/TetArg_Habitat"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetArg_Cant"
    android:layout_alignLeft="@+id/TetArg_Habitat"
    android:layout_alignRight="@+id/TetArg_Habitat"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/TetArg_TextHabitat"
    android:background="#c6c6c6">

    <TextView android:text="@string/Cant"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetArg_TitolCant"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/CantTettigettalnaArgentata"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TetArg_TextCant"
    android:layout_below="@+id/TetArg_Cant"
    android:layout_alignLeft="@+id/TetArg_Cant"
    android:layout_alignRight="@+id/TetArg_Cant"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/TetArg_Cant1"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/tettigettalna_argentata_cant1"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TetArg_TextCant"
    android:layout_alignLeft="@+id/TetArg_Cant"
    android:layout_alignRight="@+id/TetArg_Cant"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TetArg_Cant1_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TetArg_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/TetArg_Cant1_Pausa"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/pausa_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/TetArg_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"
        android:visibility="gone"/>

    <TextView android:text=""
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetArg_EspaiBlanc"
        android:layout_below="@+id/TetArg_Cant1"
        android:layout_alignLeft="@+id/TetArg_Cant1"
        android:layout_alignRight="@+id/TetArg_Cant1"
        android:layout_marginTop="5dp"
        android:textSize="12sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>
</ScrollView>
```

fitxa_tettigettula_pygmea.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#ffffff">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent" android:layout_height="match_parent">

        <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="match_parent"
            android:layout_height="60dp"
            android:paddingLeft="0dp"
            android:paddingRight="@dimen/activity_horizontal_margin"
            android:paddingTop="0dp"
            android:paddingBottom="0dp"
            android:id="@+id/TetPyg_Titol"
            android:layout_marginTop="0dp"
            android:layout_marginLeft="0dp"
            android:layout_marginRight="0dp"
            android:background="@drawable/degradat_blau">

            <TextView android:text="@string/TettigettulaPygmea"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textColor="#ffffff"
                android:id="@+id/TetPyg_NomEspecie"
                android:layout_centerVertical="true"
                android:gravity="center_horizontal"
                android:layout_centerHorizontal="true"
                android:textSize="26sp"/>
        </RelativeLayout>

        <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingLeft="@dimen/activity_horizontal_margin"
            android:paddingRight="@dimen/activity_horizontal_margin"
            android:paddingTop="0dp"
            android:paddingBottom="0dp"
            android:id="@+id/TetPyg_Imatge"
            android:layout_marginTop="0dp"
            android:layout_marginLeft="0dp"
            android:layout_marginRight="0dp"
            android:layout_below="@+id/TetPyg_Titol"
            android:background="#ffffff">

            <ImageView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/TetPyg_ImatgeEspecie"
                android:layout_centerHorizontal="true"
                android:layout_centerVertical="false"
                android:src="@drawable/tettigettula_pygmea_370"
                android:adjustViewBounds="true"
                android:layout_marginBottom="5dp"
                android:layout_marginTop="5dp"/>
```

```
    <TextView android:text="@string/ThomasHertach"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetPyg_AutoriaFotografia"
        android:layout_below="@+id/TetPyg_ImatgeEspecie"
        android:layout_alignRight="@+id/TetPyg_ImatgeEspecie"
        android:textSize="8sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetPyg_Fenologia"
    android:layout_alignLeft="@+id/TetPyg_Imatge"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_below="@+id/TetPyg_Imatge"
    android:background="#c6c6c6">

    <TextView android:text="@string/Fenologia"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetPyg_TitolFenologia"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/TetPyg_ImatgeFenologia"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/fenologia_tettigettula_pygmea"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TetPyg_Fenologia"
    android:layout_alignLeft="@+id/TetPyg_Fenologia"
    android:layout_alignRight="@+id/TetPyg_Fenologia"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetPyg_Identificacio"
    android:layout_alignLeft="@+id/TetPyg_Fenologia"
    android:layout_alignRight="@+id/TetPyg_Fenologia"
    android:layout_marginTop="15dp"
    android:layout_below="@+id/TetPyg_ImatgeFenologia"
    android:background="#c6c6c6">

    <TextView android:text="@string/Identificacio"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetPyg_TitolIdentificacio"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioTettigettulaPygmea"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TetPyg_TextIdentificacio"
    android:layout_below="@+id/TetPyg_Identificacio"
    android:layout_alignLeft="@+id/TetPyg_Identificacio"
    android:layout_alignRight="@+id/TetPyg_Identificacio"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="120dp"
    android:layout_height="match_parent"
    android:id="@+id/TetPyg_ImatgeIdentificacio"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/identificacio_tettigettula_pygmea"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TetPyg_TextIdentificacio"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="5dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetPyg_Habitat"
    android:layout_alignLeft="@+id/TetPyg_Identificacio"
    android:layout_alignRight="@+id/TetPyg_Identificacio"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/TetPyg_ImatgeIdentificacio"
    android:background="#c6c6c6">

    <TextView android:text="@string/Habitat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetPyg_TitolHabitat"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/HabitatTettigettulaPygmea"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TetPyg_TextHabitat"
    android:layout_below="@+id/TetPyg_Habitat"
    android:layout_alignLeft="@+id/TetPyg_Habitat"
    android:layout_alignRight="@+id/TetPyg_Habitat"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TetPyg_Cant"
    android:layout_alignLeft="@+id/TetPyg_Habitat"
    android:layout_alignRight="@+id/TetPyg_Habitat"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/TetPyg_TextHabitat"
    android:background="#c6c6c6">

    <TextView android:text="@string/Cant"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetPyg_TitolCant"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/CantTettigettulaPygmea"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TetPyg_TextCant"
    android:layout_below="@+id/TetPyg_Cant"
    android:layout_alignLeft="@+id/TetPyg_Cant"
    android:layout_alignRight="@+id/TetPyg_Cant"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
```

```
        android:id="@+id/TetPyg_Cant1"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/tettigettula_pygmea_cant1"
        android:layout_marginBottom="5dp"
        android:layout_below="@+id/TetPyg_TextCant"
        android:layout_alignLeft="@+id/TetPyg_Cant"
        android:layout_alignRight="@+id/TetPyg_Cant"
        android:layout_marginTop="15dp"
        android:layout_marginLeft="15dp"
        android:layout_marginRight="15dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/TetPyg_Cant1_Play"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/play_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/TetPyg_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/TetPyg_Cant1_Pausa"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/pausa_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/TetPyg_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"
        android:visibility="gone"/>

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:id="@+id/TetPyg_Cant2"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/tettigettula_pygmea_cant2"
        android:layout_marginBottom="5dp"
        android:layout_below="@+id/TetPyg_Cant1"
        android:layout_alignLeft="@+id/TetPyg_Cant1"
        android:layout_alignRight="@+id/TetPyg_Cant1"
        android:layout_marginTop="5dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/TetPyg_Cant2_Play"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/play_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/TetPyg_Cant2"
        android:layout_marginTop="10dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/TetPyg_Cant2_Pausa"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/pausa_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/TetPyg_Cant2"
        android:layout_marginTop="10dp"
        android:textSize="12sp"
        android:visibility="gone"/>

    <TextView android:text=""
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TetPyg_EspaiBlanc"
        android:layout_below="@+id/TetPyg_Cant2"
        android:layout_alignLeft="@+id/TetPyg_Cant2"
        android:layout_alignRight="@+id/TetPyg_Cant2"
        android:layout_marginTop="5dp"
        android:textSize="12sp"
        android:layout_marginLeft="5dp"/>
  </RelativeLayout>
</ScrollView>
```

fitxa_tibicina_corsica_fairmairei.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="#ffffff">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent" android:layout_height="match_parent">

        <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="match_parent"
            android:layout_height="60dp"
            android:paddingLeft="0dp"
            android:paddingRight="@dimen/activity_horizontal_margin"
            android:paddingTop="0dp"
            android:paddingBottom="0dp"
            android:id="@+id/TibCor_Titol"
            android:layout_marginTop="0dp"
            android:layout_marginLeft="0dp"
            android:layout_marginRight="0dp"
            android:background="@drawable/degradat_blau">

            <TextView android:text="@string/TibicinaCorsicaFairmairei"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textColor="#ffffff"
                android:id="@+id/TibCor_NomEspecie"
                android:layout_centerVertical="true"
                android:gravity="center_horizontal"
                android:layout_centerHorizontal="true"
                android:textSize="25sp"/>
        </RelativeLayout>

        <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingLeft="@dimen/activity_horizontal_margin"
            android:paddingRight="@dimen/activity_horizontal_margin"
            android:paddingTop="0dp"
            android:paddingBottom="0dp"
            android:id="@+id/TibCor_Imatge"
            android:layout_marginTop="0dp"
            android:layout_marginLeft="0dp"
            android:layout_marginRight="0dp"
            android:layout_below="@+id/TibCor_Titol"
            android:background="#ffffff">

            <ImageView
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:id="@+id/TibCor_ImatgeEspecie"
                android:layout_centerHorizontal="true"
                android:layout_centerVertical="false"
                android:src="@drawable/tibicina_corsica_fairmairei_370"
                android:adjustViewBounds="true"
                android:layout_marginBottom="5dp"
                android:layout_marginTop="5dp"/>

            <TextView android:text="@string/JeromeSueur"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textColor="#000000"
                android:id="@+id/TibCor_AutoriaFotografia"
                android:layout_below="@+id/TibCor_ImatgeEspecie"
                android:layout_alignRight="@+id/TibCor_ImatgeEspecie"
                android:textSize="8sp"/>
        </RelativeLayout>

        <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:paddingLeft="0dp"
            android:paddingRight="0dp"
            android:paddingTop="0dp"
            android:paddingBottom="0dp"
            android:id="@+id/TibCor_Fenologia"
            android:layout_alignLeft="@+id/TibCor_Imatge"
            android:layout_marginTop="15dp"
            android:layout_marginLeft="20dp"
            android:layout_marginRight="20dp"
            android:layout_below="@+id/TibCor_Imatge"
            android:background="#c6c6c6">

            <TextView android:text="@string/Fenologia"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
```

```
      android:textColor="#000000"
      android:id="@+id/TibCor_TitolFenologia"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<ImageView
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:id="@+id/TibCor_ImatgeFenologia"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/fenologia_tibicina_corsica_fairmairei"
   android:adjustViewBounds="true"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/TibCor_Fenologia"
   android:layout_alignLeft="@+id/TibCor_Fenologia"
   android:layout_alignRight="@+id/TibCor_Fenologia"
   android:layout_marginTop="10dp"
   android:layout_marginLeft="15dp"
   android:layout_marginRight="15dp"
   android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/TibCor_Identificacio"
   android:layout_alignLeft="@+id/TibCor_Fenologia"
   android:layout_alignRight="@+id/TibCor_Fenologia"
   android:layout_marginTop="15dp"
   android:layout_below="@+id/TibCor_ImatgeFenologia"
   android:background="#c6c6c6">

   <TextView android:text="@string/Identificacio"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibCor_TitolIdentificacio"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioTibicinaCorsicaFairmairei"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/TibCor_TextIdentificacio"
   android:layout_below="@+id/TibCor_Identificacio"
   android:layout_alignLeft="@+id/TibCor_Identificacio"
   android:layout_alignRight="@+id/TibCor_Identificacio"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>

<ImageView
   android:layout_width="190dp"
   android:layout_height="match_parent"
   android:id="@+id/TibCor_ImatgeIdentificacio"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/identificacio_tibicina_corsica_fairmairei"
   android:adjustViewBounds="true"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/TibCor_TextIdentificacio"
   android:layout_marginTop="10dp"
   android:layout_marginLeft="5dp"
   android:layout_marginRight="15dp"
   android:textSize="12sp"/>

<TextView android:text="@string/MichelBoulard"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/TibCor_AutoriaFotografiaIdentificacio"
   android:layout_below="@+id/TibCor_ImatgeIdentificacio"
   android:layout_alignLeft="@+id/TibCor_ImatgeIdentificacio"
   android:layout_alignRight="@+id/TibCor_ImatgeIdentificacio"
   android:gravity="right"
   android:textSize="7sp"
   android:layout_marginLeft="0dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/TibCor_Habitat"
   android:layout_alignLeft="@+id/TibCor_Identificacio"
   android:layout_alignRight="@+id/TibCor_Identificacio"
   android:layout_marginTop="17dp"
   android:layout_below="@+id/TibCor_ImatgeIdentificacio"
   android:background="#c6c6c6">

   <TextView android:text="@string/Habitat"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibCor_TitolHabitat"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/HabitatTibicinaCorsicaFairmairei"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/TibCor_TextHabitat"
   android:layout_below="@+id/TibCor_Habitat"
   android:layout_alignLeft="@+id/TibCor_Habitat"
   android:layout_alignRight="@+id/TibCor_Habitat"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/TibCor_Cant"
   android:layout_alignLeft="@+id/TibCor_Habitat"
   android:layout_alignRight="@+id/TibCor_Habitat"
   android:layout_marginTop="17dp"
   android:layout_below="@+id/TibCor_TextHabitat"
   android:background="#c6c6c6">

   <TextView android:text="@string/Cant"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibCor_TitolCant"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/CantTibicinaCorsicaFairmairei"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/TibCor_TextCant"
   android:layout_below="@+id/TibCor_Cant"
   android:layout_alignLeft="@+id/TibCor_Cant"
   android:layout_alignRight="@+id/TibCor_Cant"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>

<ImageView
   android:layout_width="match_parent"
   android:layout_height="45dp"
   android:id="@+id/TibCor_Cant1"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/tibicina_corsica_fairmairei_cant1"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/TibCor_TextCant"
   android:layout_alignLeft="@+id/TibCor_Cant"
   android:layout_alignRight="@+id/TibCor_Cant"
   android:layout_marginTop="15dp"
   android:layout_marginLeft="15dp"
   android:layout_marginRight="15dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
```

```
    android:id="@+id/TibCor_Cant1_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibCor_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibCor_Cant1_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibCor_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/TibCor_Cant2"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/tibicina_corsica_fairmairei_cant2"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TibCor_Cant1"
    android:layout_alignLeft="@+id/TibCor_Cant1"
    android:layout_alignRight="@+id/TibCor_Cant1"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibCor_Cant2_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibCor_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibCor_Cant2_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibCor_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/TibCor_Cant3"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/tibicina_corsica_fairmairei_cant3"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TibCor_Cant2"
    android:layout_alignLeft="@+id/TibCor_Cant2"
    android:layout_alignRight="@+id/TibCor_Cant2"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibCor_Cant3_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibCor_Cant3"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibCor_Cant3_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibCor_Cant3"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>


  <TextView android:text=""
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TibCor_EspaiBlanc"
    android:layout_below="@+id/TibCor_Cant3"
    android:layout_alignLeft="@+id/TibCor_Cant3"
    android:layout_alignRight="@+id/TibCor_Cant3"
    android:layout_marginTop="5dp"
    android:textSize="12sp"
    android:layout_marginLeft="5dp"/>
 </RelativeLayout>
</ScrollView>
```

fitxa_tibicina_garricola.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibGar_Titol"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/TibicinaGarricola"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/TibGar_NomEspecie"
        android:layout_centerVertical="true"
        android:gravity="center_horizontal"
        android:layout_centerHorizontal="true"
        android:textSize="26sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibGar_Imatge"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:layout_below="@+id/TibGar_Titol"
      android:background="#ffffff">

      <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/TibGar_ImatgeEspecie"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/tibicina_garricola_370"
        android:adjustViewBounds="true"
        android:layout_marginBottom="5dp"
        android:layout_marginTop="5dp"/>

      <TextView android:text="@string/StephanePuissant"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
```

```
    android:id="@+id/TibGar_AutoriaFotografia"
    android:layout_below="@+id/TibGar_ImatgeEspecie"
    android:layout_alignRight="@+id/TibGar_ImatgeEspecie"
    android:textSize="8sp"/>
</RelativeLayout>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/TibGar_Fenologia"
  android:layout_alignLeft="@+id/TibGar_Imatge"
  android:layout_marginTop="15dp"
  android:layout_marginLeft="20dp"
  android:layout_marginRight="20dp"
  android:layout_below="@+id/TibGar_Imatge"
  android:background="#c6c6c6">

  <TextView android:text="@string/Fenologia"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TibGar_TitolFenologia"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<ImageView
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:id="@+id/TibGar_ImatgeFenologia"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/fenologia_tibicina_garricola"
  android:adjustViewBounds="true"
  android:layout_marginBottom="5dp"
  android:layout_below="@+id/TibGar_Fenologia"
  android:layout_alignLeft="@+id/TibGar_Fenologia"
  android:layout_alignRight="@+id/TibGar_Fenologia"
  android:layout_marginTop="10dp"
  android:layout_marginLeft="15dp"
  android:layout_marginRight="15dp"
  android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/TibGar_Identificacio"
  android:layout_alignLeft="@+id/TibGar_Fenologia"
  android:layout_alignRight="@+id/TibGar_Fenologia"
  android:layout_marginTop="15dp"
  android:layout_below="@+id/TibGar_ImatgeFenologia"
  android:background="#c6c6c6">

  <TextView android:text="@string/Identificacio"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TibGar_TitolIdentificacio"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioTibicinaGarricola"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/TibGar_TextIdentificacio"
  android:layout_below="@+id/TibGar_Identificacio"
  android:layout_alignLeft="@+id/TibGar_Identificacio"
  android:layout_alignRight="@+id/TibGar_Identificacio"
  android:layout_marginTop="8dp"
  android:textSize="13sp"
  android:layout_marginLeft="5dp"/>

<ImageView
  android:layout_width="120dp"
  android:layout_height="match_parent"
  android:id="@+id/TibGar_ImatgeIdentificacio"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/identificacio_tibicina_garricola"
  android:adjustViewBounds="true"
  android:layout_marginBottom="5dp"
  android:layout_below="@+id/TibGar_TextIdentificacio"
  android:layout_marginTop="10dp"
  android:layout_marginLeft="5dp"
  android:layout_marginRight="15dp"
  android:textSize="12sp"/>

<TextView android:text="@string/VincentDerreumaux"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/TibGar_AutoriaFotografiaIdentificacio"
  android:layout_below="@+id/TibGar_ImatgeIdentificacio"
  android:layout_alignLeft="@+id/TibGar_ImatgeIdentificacio"
  android:layout_alignRight="@+id/TibGar_ImatgeIdentificacio"
  android:gravity="right"
  android:textSize="7sp"
  android:layout_marginLeft="0dp"/>


<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/TibGar_Habitat"
  android:layout_alignLeft="@+id/TibGar_Identificacio"
  android:layout_alignRight="@+id/TibGar_Identificacio"
  android:layout_marginTop="17dp"
  android:layout_below="@+id/TibGar_AutoriaFotografiaIdentificacio"
  android:background="#c6c6c6">

  <TextView android:text="@string/Habitat"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TibGar_TitolHabitat"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/HabitatTibicinaGarricola"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/TibGar_TextHabitat"
  android:layout_below="@+id/TibGar_Habitat"
  android:layout_alignLeft="@+id/TibGar_Habitat"
  android:layout_alignRight="@+id/TibGar_Habitat"
  android:layout_marginTop="8dp"
  android:textSize="13sp"
  android:layout_marginLeft="5dp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/TibGar_Cant"
  android:layout_alignLeft="@+id/TibGar_Habitat"
  android:layout_alignRight="@+id/TibGar_Habitat"
  android:layout_marginTop="17dp"
  android:layout_below="@+id/TibGar_TextHabitat"
  android:background="#c6c6c6">

  <TextView android:text="@string/Cant"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TibGar_TitolCant"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/CantTibicinaGarricola"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/TibGar_TextCant"
  android:layout_below="@+id/TibGar_Cant"
  android:layout_alignLeft="@+id/TibGar_Cant"
```

```
        android:layout_alignRight="@+id/TibGar_Cant"
        android:layout_marginTop="8dp"
        android:textSize="13sp"
        android:layout_marginLeft="5dp"/>

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="45dp"
        android:id="@+id/TibGar_Cant1"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/tibicina_garricola_cant1"
        android:layout_marginBottom="5dp"
        android:layout_below="@+id/TibGar_TextCant"
        android:layout_alignLeft="@+id/TibGar_Cant"
        android:layout_alignRight="@+id/TibGar_Cant"
        android:layout_marginTop="15dp"
        android:layout_marginLeft="15dp"
        android:layout_marginRight="15dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/TibGar_Cant1_Play"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/play_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/TibGar_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"/>

    <ImageView
        android:layout_width="25dp"
        android:layout_height="25dp"
        android:id="@+id/TibGar_Cant1_Pausa"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/pausa_gris"
        android:layout_marginBottom="5dp"
        android:layout_alignTop="@+id/TibGar_Cant1"
        android:layout_marginTop="10dp"
        android:textSize="12sp"
        android:visibility="gone"/>

    <TextView android:text="@string/FotografiaResultatsVincentDerreumaux"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibGar_AutoriaFotografiaResultats"
        android:layout_below="@+id/TibGar_Cant1"
        android:layout_alignLeft="@+id/TibGar_Cant"
        android:layout_alignRight="@+id/TibGar_Cant"
        android:layout_marginTop="5dp"
        android:gravity="right"
        android:textSize="8sp"
        android:layout_marginLeft="0dp"/>

    <TextView android:text=""
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibGar_EspaiBlanc"
        android:layout_below="@+id/TibGar_AutoriaFotografiaResultats"
        android:layout_alignLeft="@+id/TibGar_Cant1"
        android:layout_alignRight="@+id/TibGar_Cant1"
        android:layout_marginTop="5dp"
        android:textSize="12sp"
        android:layout_marginLeft="5dp"/>
  </RelativeLayout>
</ScrollView>
```

fitxa_tibicina_haematodes.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibHae_Titol"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/TibicinaHaematodes"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/TibHae_NomEspecie"
        android:layout_centerVertical="true"
        android:gravity="center_horizontal"
        android:layout_centerHorizontal="true"
        android:textSize="26sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibHae_Imatge"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:layout_below="@+id/TibHae_Titol"
      android:background="#ffffff">

      <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/TibHae_ImatgeEspecie"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/tibicina_haematodes_370"
        android:adjustViewBounds="true"
        android:layout_marginBottom="5dp"
        android:layout_marginTop="5dp"/>

      <TextView android:text="@string/CosminOvidiu"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibHae_AutoriaFotografia"
        android:layout_below="@+id/TibHae_ImatgeEspecie"
        android:layout_alignRight="@+id/TibHae_ImatgeEspecie"
        android:textSize="8sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibHae_Fenologia"
      android:layout_alignLeft="@+id/TibHae_Imatge"
      android:layout_marginTop="15dp"
      android:layout_marginLeft="20dp"
      android:layout_marginRight="20dp"
      android:layout_below="@+id/TibHae_Imatge"
      android:background="#c6c6c6">

      <TextView android:text="@string/Fenologia"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibHae_TitolFenologia"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>

    <ImageView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/TibHae_ImatgeFenologia"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/fenologia_tibicina_haematodes"
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_below="@+id/TibHae_Fenologia"
      android:layout_alignLeft="@+id/TibHae_Fenologia"
      android:layout_alignRight="@+id/TibHae_Fenologia"
```

```
   android:layout_marginTop="10dp"
   android:layout_marginLeft="15dp"
   android:layout_marginRight="15dp"
   android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/TibHae_Identificacio"
   android:layout_alignLeft="@+id/TibHae_Fenologia"
   android:layout_alignRight="@+id/TibHae_Fenologia"
   android:layout_marginTop="15dp"
   android:layout_below="@+id/TibHae_ImatgeFenologia"
   android:background="#c6c6c6">

   <TextView android:text="@string/Identificacio"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibHae_TitolIdentificacio"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioTibicinaHaematodes"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/TibHae_TextIdentificacio"
   android:layout_below="@+id/TibHae_Identificacio"
   android:layout_alignLeft="@+id/TibHae_Identificacio"
   android:layout_alignRight="@+id/TibHae_Identificacio"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/TibHae_Habitat"
   android:layout_alignLeft="@+id/TibHae_Identificacio"
   android:layout_alignRight="@+id/TibHae_Identificacio"
   android:layout_marginTop="17dp"
   android:layout_below="@+id/TibHae_TextIdentificacio"
   android:background="#c6c6c6">

   <TextView android:text="@string/Habitat"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibHae_TitolHabitat"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/HabitatTibicinaHaematodes"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/TibHae_TextHabitat"
   android:layout_below="@+id/TibHae_Habitat"
   android:layout_alignLeft="@+id/TibHae_Habitat"
   android:layout_alignRight="@+id/TibHae_Habitat"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/TibHae_Cant"
   android:layout_alignLeft="@+id/TibHae_Habitat"
   android:layout_alignRight="@+id/TibHae_Habitat"
      android:layout_marginTop="17dp"
      android:layout_below="@+id/TibHae_TextHabitat"
      android:background="#c6c6c6">

      <TextView android:text="@string/Cant"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:textColor="#000000"
         android:id="@+id/TibHae_TitolCant"
         android:textSize="15sp"
         android:layout_marginLeft="5dp"/>
   </RelativeLayout>



   <TextView android:text="@string/CantTibicinaHaematodes"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibHae_TextCant"
      android:layout_below="@+id/TibHae_Cant"
      android:layout_alignLeft="@+id/TibHae_Cant"
      android:layout_alignRight="@+id/TibHae_Cant"
      android:layout_marginTop="8dp"
      android:textSize="13sp"
      android:layout_marginLeft="5dp"/>

   <ImageView
      android:layout_width="match_parent"
      android:layout_height="45dp"
      android:id="@+id/TibHae_Cant1"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/tibicina_haematodes_cant1"
      android:layout_marginBottom="5dp"
      android:layout_below="@+id/TibHae_TextCant"
      android:layout_alignLeft="@+id/TibHae_Cant"
      android:layout_alignRight="@+id/TibHae_Cant"
      android:layout_marginTop="15dp"
      android:layout_marginLeft="15dp"
      android:layout_marginRight="15dp"
      android:textSize="12sp"/>

   <ImageView
      android:layout_width="25dp"
      android:layout_height="25dp"
      android:id="@+id/TibHae_Cant1_Play"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/play_gris"
      android:layout_marginBottom="5dp"
      android:layout_alignTop="@+id/TibHae_Cant1"
      android:layout_marginTop="10dp"
      android:textSize="12sp"/>

   <ImageView
      android:layout_width="25dp"
      android:layout_height="25dp"
      android:id="@+id/TibHae_Cant1_Pausa"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/pausa_gris"
      android:layout_marginBottom="5dp"
      android:layout_alignTop="@+id/TibHae_Cant1"
      android:layout_marginTop="10dp"
      android:textSize="12sp"
      android:visibility="gone"/>

   <TextView android:text=""
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibHae_EspaiBlanc"
      android:layout_below="@+id/TibHae_Cant1"
      android:layout_alignLeft="@+id/TibHae_Cant1"
      android:layout_alignRight="@+id/TibHae_Cant1"
      android:layout_marginTop="5dp"
      android:textSize="12sp"
      android:layout_marginLeft="5dp"/>
   </RelativeLayout>
</ScrollView>
```

fitxa_tibicina_quadrisignata.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
```

```
android:background="#ffffff">

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent" android:layout_height="match_parent">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="60dp"
    android:paddingLeft="0dp"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TibQua_Titol"
    android:layout_marginTop="0dp"
    android:layout_marginLeft="0dp"
    android:layout_marginRight="0dp"
    android:background="@drawable/degradat_blau">

    <TextView android:text="@string/TibicinaQuadrisignata"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#ffffff"
      android:id="@+id/TibQua_NomEspecie"
      android:layout_centerVertical="true"
      android:gravity="center_horizontal"
      android:layout_centerHorizontal="true"
      android:textSize="25sp"/>
  </RelativeLayout>

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TibQua_Imatge"
    android:layout_marginTop="0dp"
    android:layout_marginLeft="0dp"
    android:layout_marginRight="0dp"
    android:layout_below="@+id/TibQua_Titol"
    android:background="#ffffff">

    <ImageView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/TibQua_ImatgeEspecie"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/tibicina_quadrisignata_370"
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_marginTop="5dp"/>

    <TextView android:text="@string/PerePons"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibQua_AutoriaFotografia"
      android:layout_below="@+id/TibQua_ImatgeEspecie"
      android:layout_alignRight="@+id/TibQua_ImatgeEspecie"
      android:textSize="8sp"/>
  </RelativeLayout>

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TibQua_Fenologia"
    android:layout_alignLeft="@+id/TibQua_Imatge"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="20dp"
    android:layout_marginRight="20dp"
    android:layout_below="@+id/TibQua_Imatge"
    android:background="#c6c6c6">

    <TextView android:text="@string/Fenologia"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibQua_TitolFenologia"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
  </RelativeLayout>

  <ImageView
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:id="@+id/TibQua_ImatgeFenologia"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/fenologia_tibicina_quadrisignata"
    android:adjustViewBounds="true"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TibQua_Fenologia"
    android:layout_alignLeft="@+id/TibQua_Fenologia"
    android:layout_alignRight="@+id/TibQua_Fenologia"
    android:layout_marginTop="10dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/TibQua_Identificacio"
  android:layout_alignLeft="@+id/TibQua_Fenologia"
  android:layout_alignRight="@+id/TibQua_Fenologia"
  android:layout_marginTop="15dp"
  android:layout_below="@+id/TibQua_ImatgeFenologia"
  android:background="#c6c6c6">

  <TextView android:text="@string/Identificacio"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TibQua_TitolIdentificacio"
    android:textSize="15sp"
    android:layout_marginLeft="5dp"/>
</RelativeLayout>

<TextView android:text="@string/IdentificacioTibicinaQuadrisignata"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/TibQua_TextIdentificacio"
  android:layout_below="@+id/TibQua_Identificacio"
  android:layout_alignLeft="@+id/TibQua_Identificacio"
  android:layout_alignRight="@+id/TibQua_Identificacio"
  android:layout_marginTop="8dp"
  android:textSize="13sp"
  android:layout_marginLeft="5dp"/>

<ImageView
  android:layout_width="95dp"
  android:layout_height="match_parent"
  android:id="@+id/TibQua_ImatgeIdentificacio"
  android:layout_centerHorizontal="true"
  android:layout_centerVertical="false"
  android:src="@drawable/identificacio_tibicina_quadrisignata"
  android:adjustViewBounds="true"
  android:layout_marginBottom="5dp"
  android:layout_below="@+id/TibQua_TextIdentificacio"
  android:layout_marginTop="10dp"
  android:layout_marginLeft="5dp"
  android:layout_marginRight="15dp"
  android:textSize="12sp"/>

<TextView android:text="@string/PerePons"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textColor="#000000"
  android:id="@+id/TibQua_AutoriaFotografiaIdentificacio"
  android:layout_below="@+id/TibQua_ImatgeIdentificacio"
  android:layout_alignLeft="@+id/TibQua_ImatgeIdentificacio"
  android:layout_alignRight="@+id/TibQua_ImatgeIdentificacio"
  android:gravity="right"
  android:textSize="7sp"
  android:layout_marginLeft="0dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:paddingLeft="0dp"
  android:paddingRight="0dp"
  android:paddingTop="0dp"
  android:paddingBottom="0dp"
  android:id="@+id/TibQua_Habitat"
  android:layout_alignLeft="@+id/TibQua_Identificacio"
  android:layout_alignRight="@+id/TibQua_Identificacio"
  android:layout_marginTop="17dp"
  android:layout_below="@+id/TibQua_AutoriaFotografiaIdentificacio"
  android:background="#c6c6c6">
```

```
    <TextView android:text="@string/Habitat"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibQua_TitolHabitat"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/HabitatTibicinaQuadrisignata"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TibQua_TextHabitat"
    android:layout_below="@+id/TibQua_Habitat"
    android:layout_alignLeft="@+id/TibQua_Habitat"
    android:layout_alignRight="@+id/TibQua_Habitat"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="0dp"
    android:paddingRight="0dp"
    android:paddingTop="0dp"
    android:paddingBottom="0dp"
    android:id="@+id/TibQua_Cant"
    android:layout_alignLeft="@+id/TibQua_Habitat"
    android:layout_alignRight="@+id/TibQua_Habitat"
    android:layout_marginTop="17dp"
    android:layout_below="@+id/TibQua_TextHabitat"
    android:background="#c6c6c6">

    <TextView android:text="@string/Cant"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibQua_TitolCant"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/CantTibicinaQuadrisignata"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:id="@+id/TibQua_TextCant"
    android:layout_below="@+id/TibQua_Cant"
    android:layout_alignLeft="@+id/TibQua_Cant"
    android:layout_alignRight="@+id/TibQua_Cant"
    android:layout_marginTop="8dp"
    android:textSize="13sp"
    android:layout_marginLeft="5dp"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/TibQua_Cant1"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/tibicina_quadrisignata_cant1"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TibQua_TextCant"
    android:layout_alignLeft="@+id/TibQua_Cant"
    android:layout_alignRight="@+id/TibQua_Cant"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="15dp"
    android:layout_marginRight="15dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibQua_Cant1_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibQua_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibQua_Cant1_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibQua_Cant1"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/TibQua_Cant2"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/tibicina_quadrisignata_cant2"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TibQua_Cant1"
    android:layout_alignLeft="@+id/TibQua_Cant1"
    android:layout_alignRight="@+id/TibQua_Cant1"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibQua_Cant2_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibQua_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibQua_Cant2_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibQua_Cant2"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>

<ImageView
    android:layout_width="match_parent"
    android:layout_height="45dp"
    android:id="@+id/TibQua_Cant3"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/tibicina_quadrisignata_cant3"
    android:layout_marginBottom="5dp"
    android:layout_below="@+id/TibQua_Cant2"
    android:layout_alignLeft="@+id/TibQua_Cant2"
    android:layout_alignRight="@+id/TibQua_Cant2"
    android:layout_marginTop="5dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibQua_Cant3_Play"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/play_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibQua_Cant3"
    android:layout_marginTop="10dp"
    android:textSize="12sp"/>

<ImageView
    android:layout_width="25dp"
    android:layout_height="25dp"
    android:id="@+id/TibQua_Cant3_Pausa"
    android:layout_centerHorizontal="true"
    android:layout_centerVertical="false"
    android:src="@drawable/pausa_gris"
    android:layout_marginBottom="5dp"
    android:layout_alignTop="@+id/TibQua_Cant3"
    android:layout_marginTop="10dp"
    android:textSize="12sp"
    android:visibility="gone"/>
```

```
    <TextView android:text=""
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibQua_EspaiBlanc"
      android:layout_below="@+id/TibQua_Cant3"
      android:layout_alignLeft="@+id/TibQua_Cant3"
      android:layout_alignRight="@+id/TibQua_Cant3"
      android:layout_marginTop="5dp"
      android:textSize="12sp"
      android:layout_marginLeft="5dp"/>
  </RelativeLayout>
</ScrollView>
```

fitxa_tibicina_tomentosa.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:background="#ffffff">

  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent" android:layout_height="match_parent">

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="60dp"
      android:paddingLeft="0dp"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibTom_Titol"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:background="@drawable/degradat_blau">

      <TextView android:text="@string/TibicinaTomentosa"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#ffffff"
        android:id="@+id/TibTom_NomEspecie"
        android:layout_centerVertical="true"
        android:gravity="center_horizontal"
        android:layout_centerHorizontal="true"
        android:textSize="25sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="@dimen/activity_horizontal_margin"
      android:paddingRight="@dimen/activity_horizontal_margin"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibTom_Imatge"
      android:layout_marginTop="0dp"
      android:layout_marginLeft="0dp"
      android:layout_marginRight="0dp"
      android:layout_below="@+id/TibTom_Titol"
      android:background="#ffffff">

      <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:id="@+id/TibTom_ImatgeEspecie"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="false"
        android:src="@drawable/tibicina_tomentosa_370"
        android:adjustViewBounds="true"
        android:layout_marginBottom="5dp"
        android:layout_marginTop="5dp"/>

      <TextView android:text="@string/IgnasiJosepTejedor"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibTom_AutoriaFotografia"
        android:layout_below="@+id/TibTom_ImatgeEspecie"
        android:layout_alignRight="@+id/TibTom_ImatgeEspecie"
        android:textSize="8sp"/>
    </RelativeLayout>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibTom_Fenologia"
      android:layout_alignLeft="@+id/TibTom_Imatge"
      android:layout_marginTop="15dp"
      android:layout_marginLeft="20dp"
      android:layout_marginRight="20dp"
      android:layout_below="@+id/TibTom_Imatge"
      android:background="#c6c6c6">

      <TextView android:text="@string/Fenologia"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibTom_TitolFenologia"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>

    <ImageView
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:id="@+id/TibTom_ImatgeFenologia"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="false"
      android:src="@drawable/fenologia_tibicina_tomentosa"
      android:adjustViewBounds="true"
      android:layout_marginBottom="5dp"
      android:layout_below="@+id/TibTom_Fenologia"
      android:layout_alignLeft="@+id/TibTom_Fenologia"
      android:layout_alignRight="@+id/TibTom_Fenologia"
      android:layout_marginTop="10dp"
      android:layout_marginLeft="15dp"
      android:layout_marginRight="15dp"
      android:textSize="12sp"/>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibTom_Identificacio"
      android:layout_alignLeft="@+id/TibTom_Fenologia"
      android:layout_alignRight="@+id/TibTom_Fenologia"
      android:layout_marginTop="15dp"
      android:layout_below="@+id/TibTom_ImatgeFenologia"
      android:background="#c6c6c6">

      <TextView android:text="@string/Identificacio"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibTom_TitolIdentificacio"
        android:textSize="15sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>

    <TextView android:text="@string/IdentificacioTibicinaTomentosa"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibTom_TextIdentificacio"
      android:layout_below="@+id/TibTom_Identificacio"
      android:layout_alignLeft="@+id/TibTom_Identificacio"
      android:layout_alignRight="@+id/TibTom_Identificacio"
      android:layout_marginTop="8dp"
      android:textSize="13sp"
      android:layout_marginLeft="5dp"/>

    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
      android:layout_width="match_parent"
      android:layout_height="match_parent"
      android:paddingLeft="0dp"
      android:paddingRight="0dp"
      android:paddingTop="0dp"
      android:paddingBottom="0dp"
      android:id="@+id/TibTom_Habitat"
      android:layout_alignLeft="@+id/TibTom_Identificacio"
      android:layout_alignRight="@+id/TibTom_Identificacio"
      android:layout_marginTop="17dp"
      android:layout_below="@+id/TibTom_TextIdentificacio"
      android:background="#c6c6c6">

      <TextView android:text="@string/Habitat"
        android:layout_width="wrap_content"
```

```
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibTom_TitolHabitat"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/HabitatTibicinaTomentosa"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/TibTom_TextHabitat"
   android:layout_below="@+id/TibTom_Habitat"
   android:layout_alignLeft="@+id/TibTom_Habitat"
   android:layout_alignRight="@+id/TibTom_Habitat"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>



<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:paddingLeft="0dp"
   android:paddingRight="0dp"
   android:paddingTop="0dp"
   android:paddingBottom="0dp"
   android:id="@+id/TibTom_Cant"
   android:layout_alignLeft="@+id/TibTom_Habitat"
   android:layout_alignRight="@+id/TibTom_Habitat"
   android:layout_marginTop="17dp"
   android:layout_below="@+id/TibTom_TextHabitat"
   android:background="#c6c6c6">

   <TextView android:text="@string/Cant"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textColor="#000000"
      android:id="@+id/TibTom_TitolCant"
      android:textSize="15sp"
      android:layout_marginLeft="5dp"/>
</RelativeLayout>



<TextView android:text="@string/CantTibicinaTomentosa"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:textColor="#000000"
   android:id="@+id/TibTom_TextCant"
   android:layout_below="@+id/TibTom_Cant"
   android:layout_alignLeft="@+id/TibTom_Cant"
   android:layout_alignRight="@+id/TibTom_Cant"
   android:layout_marginTop="8dp"
   android:textSize="13sp"
   android:layout_marginLeft="5dp"/>

<ImageView
   android:layout_width="match_parent"
   android:layout_height="45dp"
   android:id="@+id/TibTom_Cant1"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/tibicina_tomentosa_cant1"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/TibTom_TextCant"
   android:layout_alignLeft="@+id/TibTom_Cant"
   android:layout_alignRight="@+id/TibTom_Cant"
   android:layout_marginTop="15dp"
   android:layout_marginLeft="15dp"
   android:layout_marginRight="15dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/TibTom_Cant1_Play"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/play_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/TibTom_Cant1"
   android:layout_marginTop="10dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/TibTom_Cant1_Pausa"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/pausa_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/TibTom_Cant1"
   android:layout_marginTop="10dp"
   android:textSize="12sp"
   android:visibility="gone"/>

<ImageView
   android:layout_width="match_parent"
   android:layout_height="45dp"
   android:id="@+id/TibTom_Cant2"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/tibicina_tomentosa_cant2"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/TibTom_Cant1"
   android:layout_alignLeft="@+id/TibTom_Cant1"
   android:layout_alignRight="@+id/TibTom_Cant1"
   android:layout_marginTop="5dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/TibTom_Cant2_Play"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/play_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/TibTom_Cant2"
   android:layout_marginTop="10dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/TibTom_Cant2_Pausa"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/pausa_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/TibTom_Cant2"
   android:layout_marginTop="10dp"
   android:textSize="12sp"
   android:visibility="gone"/>

<ImageView
   android:layout_width="match_parent"
   android:layout_height="45dp"
   android:id="@+id/TibTom_Cant3"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/tibicina_tomentosa_cant3"
   android:layout_marginBottom="5dp"
   android:layout_below="@+id/TibTom_Cant2"
   android:layout_alignLeft="@+id/TibTom_Cant2"
   android:layout_alignRight="@+id/TibTom_Cant2"
   android:layout_marginTop="5dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/TibTom_Cant3_Play"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/play_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/TibTom_Cant3"
   android:layout_marginTop="10dp"
   android:textSize="12sp"/>

<ImageView
   android:layout_width="25dp"
   android:layout_height="25dp"
   android:id="@+id/TibTom_Cant3_Pausa"
   android:layout_centerHorizontal="true"
   android:layout_centerVertical="false"
   android:src="@drawable/pausa_gris"
   android:layout_marginBottom="5dp"
   android:layout_alignTop="@+id/TibTom_Cant3"
   android:layout_marginTop="10dp"
   android:textSize="12sp"
   android:visibility="gone"/>


<TextView android:text=""
   android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:textColor="#000000"
        android:id="@+id/TibTom_EspaiBlanc"
        android:layout_below="@+id/TibTom_Cant3"
        android:layout_alignLeft="@+id/TibTom_Cant3"
        android:layout_alignRight="@+id/TibTom_Cant3"
        android:layout_marginTop="5dp"
        android:textSize="12sp"
        android:layout_marginLeft="5dp"/>
    </RelativeLayout>
</ScrollView>
```

instruccions.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">
    <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:orientation="vertical"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="#ffffff">

        <ImageView
            android:layout_width="320dp"
            android:layout_height="60dp"
            android:id="@+id/Instruccions_FonsTitol"
            android:background="#5a5a5a"/>

        <ImageView
            android:layout_marginLeft="10dp"
            android:layout_marginTop="15dp"
            android:layout_width="30dp"
            android:layout_height="30dp"
            android:id="@+id/Instruccions_ImatgeTitol"
            android:src="@drawable/info"/>

        <TextView
            android:layout_width="225dp"
            android:layout_height="60dp"
            android:textAppearance="?android:attr/textAppearanceLarge"
            android:text="@string/InstruccionsDUs"
            android:id="@+id/Instruccions_InstruccionsDUs"
            android:layout_alignParentTop="true"
            android:gravity="center_vertical"
            android:textColor="#ffffff"
            android:layout_marginLeft="55dp"
            android:textSize="24sp"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="@string/InstruccionsIConsells"
            android:id="@+id/Instruccions_InstruccionsIConsells"
            android:gravity="center_vertical"
            android:textColor="#000000"
            android:layout_marginLeft="20dp"
            android:layout_marginTop="80dp"
            android:layout_marginRight="30dp"
            android:textSize="13sp"
            android:textStyle="bold"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="@string/Instruccio1"
            android:id="@+id/Instruccions_Instruccio1"
            android:gravity="left"
            android:textColor="#000000"
            android:layout_below="@+id/Instruccions_InstruccionsIConsells"
            android:layout_marginLeft="35dp"
            android:layout_marginTop="25dp"
            android:layout_marginRight="30dp"
            android:textSize="11sp"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="1. "
            android:id="@+id/Instruccions_1"
            android:gravity="left"
            android:textColor="#000000"
            android:layout_below="@+id/Instruccions_InstruccionsIConsells"
            android:layout_marginLeft="20dp"
            android:layout_marginTop="25dp"
            android:layout_marginRight="30dp"
            android:textSize="11sp"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="2. "
            android:id="@+id/Instruccions_2"
            android:gravity="left"
            android:textColor="#000000"
            android:layout_marginLeft="20dp"
            android:layout_marginTop="20dp"
            android:layout_marginRight="30dp"
            android:textSize="11sp"
            android:layout_below="@id/Instruccions_Instruccio1"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="@string/Instruccio2"
            android:id="@+id/Instruccions_Instruccio2"
            android:gravity="left"
            android:textColor="#000000"
            android:layout_marginLeft="35dp"
            android:layout_marginTop="20dp"
            android:layout_marginRight="30dp"
            android:textSize="11sp"
            android:layout_below="@id/Instruccions_Instruccio1"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="3. "
            android:id="@+id/Instruccions_3"
            android:gravity="left"
            android:textColor="#000000"
            android:layout_below="@+id/Instruccions_Instruccio2"
            android:layout_marginLeft="20dp"
            android:layout_marginTop="20dp"
            android:layout_marginRight="30dp"
            android:textSize="11sp"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="@string/Instruccio3"
            android:id="@+id/Instruccions_Instruccio3"
            android:gravity="left"
            android:textColor="#000000"
            android:layout_below="@+id/Instruccions_Instruccio2"
            android:layout_marginLeft="35dp"
            android:layout_marginTop="20dp"
            android:layout_marginRight="30dp"
            android:textSize="11sp"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="4. "
            android:id="@+id/Instruccions_4"
            android:gravity="left"
            android:textColor="#000000"
            android:layout_below="@+id/Instruccions_Instruccio3"
            android:layout_marginLeft="20dp"
            android:layout_marginTop="20dp"
            android:layout_marginRight="30dp"
            android:textSize="11sp"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textAppearance="?android:attr/textAppearanceMedium"
            android:text="@string/Instruccio4"
            android:id="@+id/Instruccions_Instruccio4"
            android:gravity="left"
            android:textColor="#000000"
            android:layout_below="@+id/Instruccions_Instruccio3"
            android:layout_marginLeft="35dp"
            android:layout_marginTop="20dp"
            android:layout_marginRight="30dp"
            android:textSize="11sp"/>

        <TextView
            android:layout_width="wrap_content"
```

```
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/textAppearanceMedium"
      android:text="@string/Instruccio5"
      android:id="@+id/Instruccions_Instruccio5"
      android:gravity="left"
      android:textColor="#000000"
      android:layout_below="@+id/Instruccions_Instruccio4"
      android:layout_marginLeft="35dp"
      android:layout_marginTop="20dp"
      android:layout_marginRight="30dp"
      android:textSize="11sp"/>

    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/textAppearanceMedium"
      android:text="5. "
      android:id="@+id/Instruccions_5"
      android:gravity="left"
      android:textColor="#000000"
      android:layout_below="@+id/Instruccions_Instruccio4"
      android:layout_marginLeft="20dp"
      android:layout_marginTop="20dp"
      android:layout_marginRight="30dp"
      android:textSize="11sp"/>


    <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:textAppearance="?android:attr/textAppearanceMedium"
      android:text=""
      android:id="@+id/Instruccions_EspaiBlanc"
      android:gravity="center_vertical"
      android:textColor="#000000"
      android:layout_below="@+id/Instruccions_Instruccio5"
      android:layout_marginLeft="0dp"
      android:layout_marginTop="5dp"
      android:layout_marginRight="0dp"
      android:textSize="11sp"/>
  </RelativeLayout>
</ScrollView>
```

llista_configuracio.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="horizontal" >

  <LinearLayout android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <TextView
      android:id="@+id/item"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Text qualsevol"
      android:textAppearance="?android:attr/textAppearanceMedium"
      android:layout_marginLeft="4dp"
      android:layout_marginTop="4dp"
      android:padding="2dp"
      android:textColor="#000000" />
  </LinearLayout>
</LinearLayout>
```

llista_fitxers.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_height="wrap_content"          android:orientation="vertical"
android:layout_width="fill_parent">
  <ImageView
    android:id="@+id/Icona"
    android:layout_width="50dip"
    android:layout_height="50dip" >
  </ImageView>

  <TextView android:text="@+id/Text1"
    android:id="@+id/Text1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:singleLine="true"
    android:textStyle="bold"
    android:layout_toRightOf="@+id/Icona"
    android:layout_marginTop="5dip"
    android:layout_marginLeft="5dip">
  </TextView>
  <TextView android:text="@+id/Text2"
    android:id="@+id/Text2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_toRightOf="@+id/Icona"
    android:layout_below="@+id/Text1"
    android:layout_marginLeft="10dip">
  </TextView>
  <TextView android:text="@+id/Data"
    android:id="@+id/Data"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/Text1"
    android:layout_alignParentRight="true"
    android:layout_marginLeft="5dip">
  </TextView>
</RelativeLayout>
```

llista_idiomes.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="horizontal" >

  <ImageView
    android:id="@+id/icon"
    android:layout_width="30dp"
    android:layout_height="30dp"
    android:padding="5dp" />

  <LinearLayout android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <TextView
      android:id="@+id/item"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Text qualsevol"
      android:textAppearance="?android:attr/textAppearanceMedium"
      android:layout_marginLeft="3dp"
      android:layout_marginTop="2dp"
      android:padding="2dp"
      android:textColor="#000000" />
  </LinearLayout>
</LinearLayout>
```

quant_a.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView
  xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent">
  <RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:background="#ffffff">

    <ImageView
      android:layout_width="320dp"
      android:layout_height="60dp"
      android:id="@+id/QuantA_FonsTitol"
      android:background="#5a5a5a"/>

    <ImageView
      android:layout_marginLeft="10dp"
      android:layout_marginTop="15dp"
      android:layout_width="30dp"
      android:layout_height="30dp"
      android:id="@+id/QuantA_ImatgeTitol"
      android:src="@drawable/info"/>

    <TextView
      android:layout_width="225dp"
      android:layout_height="60dp"
```

```
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:text="@string/QuantA"
    android:id="@+id/QuantA_QuantA"
    android:layout_alignParentTop="true"
    android:gravity="center_vertical"
    android:textColor="#ffffff"
    android:layout_marginLeft="55dp"
    android:textSize="24sp"/>

<ImageView
    android:layout_marginLeft="20dp"
    android:layout_marginTop="80dp"
    android:layout_width="80dp"
    android:layout_height="80dp"
    android:id="@+id/QuantA_Logotip"
    android:src="@drawable/logotip"/>


<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:text="@string/app_name"
    android:id="@+id/QuantA_Titol"
    android:gravity="center_vertical"
    android:textColor="#000000"
    android:layout_alignTop="@+id/QuantA_Logotip"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="100dp"
    android:layout_marginTop="10dp"
    android:textStyle="bold"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceLarge"
    android:text="@string/Versio1.0"
    android:id="@+id/QuantA_Versio"
    android:gravity="center_vertical"
    android:textColor="#000000"
    android:layout_alignTop="@+id/QuantA_Logotip"
    android:layout_alignParentRight="true"
    android:layout_alignParentEnd="true"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="100dp"
    android:layout_marginTop="45dp"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="@string/Copyright"
    android:id="@+id/QuantA_Copyright"
    android:gravity="center_vertical"
    android:textColor="#000000"
    android:layout_alignTop="@+id/QuantA_Logotip"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="10dp"
    android:layout_marginTop="100dp"
    android:layout_marginRight="30dp"
    android:textSize="11sp"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="@string/DescripcioAplicacio"
    android:id="@+id/QuantA_Descripcio"
    android:gravity="left"
    android:textColor="#000000"
    android:layout_below="@+id/QuantA_Copyright"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="10dp"
    android:layout_marginTop="15dp"
    android:layout_marginRight="30dp"
    android:textSize="11sp"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="@string/DescripcioAplicacio2"
    android:id="@+id/QuantA_Descripcio2"
    android:gravity="left"
    android:textColor="#000000"
    android:layout_below="@+id/QuantA_Descripcio"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="10dp"
    android:layout_marginTop="10dp"
    android:layout_marginRight="30dp"
    android:textSize="11sp"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="@string/LlicenciaPart1"
    android:id="@+id/QuantA_LlicenciaPart1"
    android:gravity="left"
    android:textColor="#000000"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="10dp"
    android:layout_marginTop="20dp"
    android:layout_marginRight="30dp"
    android:textSize="11sp"
    android:layout_below="@id/QuantA_Descripcio2"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="@string/LlicenciaPart2"
    android:id="@+id/QuantA_LlicenciaPart2"
    android:gravity="left"
    android:textColor="#000000"
    android:layout_below="@+id/QuantA_LlicenciaPart1"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="10dp"
    android:layout_marginTop="10dp"
    android:layout_marginRight="30dp"
    android:textSize="11sp"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="@string/CorreuContacte"
    android:id="@+id/QuantA_CorreuContacte"
    android:gravity="left"
    android:textColor="#000000"
    android:layout_below="@+id/QuantA_LlicenciaPart2"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="10dp"
    android:layout_marginTop="10dp"
    android:layout_marginRight="30dp"
    android:textSize="11sp"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text="@string/Agraiments"
    android:id="@+id/QuantA_Agraïments"
    android:gravity="left"
    android:textColor="#000000"
    android:layout_below="@+id/QuantA_CorreuContacte"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="10dp"
    android:layout_marginTop="20dp"
    android:layout_marginRight="30dp"
    android:textSize="11sp"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textAppearance="?android:attr/textAppearanceMedium"
    android:text=""
    android:id="@+id/QuantA_EspaiBlanc"
    android:gravity="center_vertical"
    android:textColor="#000000"
    android:layout_below="@+id/QuantA_Agraïments"
    android:layout_alignLeft="@+id/QuantA_Logotip"
    android:layout_marginLeft="0dp"
    android:layout_marginTop="5dp"
    android:layout_marginRight="0dp"
    android:textSize="11sp"/>
  </RelativeLayout>
</ScrollView>
```

styles.xml:

```
<resources>

    <!-- Base application theme. -->
    <style name="AppTheme" parent="Base.Theme.AppCompat.Light.DarkActionBar">
        <!-- Customize your theme here. -->
    </style>
```

```
    <style name="AppThemeSenseActionBar" parent="Theme.AppCompat.DayNight.NoActionBar">
        <item name="android:windowActionBar">false</item>
        <item name="android:windowNoTitle">true</item>
    </style>

</resources>
```

strings.xml (cat):

```
<resources>
    <string name="app_name">CicadaApp</string>

    <string name="LesEspecies">Les espècies</string>
    <string name="IniciaUnaGravacio">Inicia una gravació</string>
    <string name="AnalitzaUnFitxer">Analitza un fitxer</string>
    <string name="title_activity_llista_especies">LlistaEspecies</string>
    <string name="title_activity_enregistrador_audio">Gravació en curs</string>
    <string name="title_activity_resultats">Resultats</string>
    <string name="title_activity_fitxa_especie">Fitxa de l\'espècie</string>
    <string name="LlistaEspecies">Llista de les espècies de cigala amb presència real o potencial a Catalunya</string>
    <string name="Catala">Català</string>
    <string name="Castella">Castellano</string>
    <string name="Angles">English</string>
    <string name="Frances">French</string>
    <string name="QuantA">Quant a CicadaApp</string>
    <string name="Instruccions">Instruccions d\'ús</string>
    <string name="Reprodueix">Reprodueix</string>
    <string name="Analitza">Analitza</string>
    <string name="Descarta">Descarta</string>
    <string name="Atura">Atura</string>
    <string name="Resultats">Resultats</string>
    <string name="ReproduccioEnCurs">Reproducció en curs</string>
    <string name="SemblancesTrobades">Semblances trobades</string>
    <string name="NoHiHaResultats">No s\'ha trobat coincidències</string>
    <string name="CicadaOrni">Cicada orni</string>
    <string name="CicadaBarbaraLusitanica">Cicada barbara lusitanica</string>
    <string name="CicadatraAtra">Cicadatra atra</string>
    <string name="CicadettaBrevipennis">Cicadetta brevipennis</string>
    <string name="CicadettaCerdaniensis">Cicadetta cerdaniensis</string>
    <string name="HilaphuraVaripes">Hilaphura varipes</string>
    <string name="LyristesPlebejus">Lyristes plebejus</string>
    <string name="TettigettalnaArgentata">Tettigettalna argentata</string>
    <string name="TettigettulaPygmea">Tettigettula pygmea</string>
    <string name="TibicinaCorsicaFairmairei">Tibicina corsica fairmairei</string>
    <string name="TibicinaGarricola">Tibicina garricola</string>
    <string name="TibicinaHaematodes">Tibicina haematodes</string>
    <string name="TibicinaQuadrisignata">Tibicina quadrisignata</string>
    <string name="TibicinaTomentosa">Tibicina tomentosa</string>
    <string name="CicadaBarbara">Cicada barbara</string>
    <string name="TibicinaCorsica">Tibicina corsica</string>
    <string name="SspLusitanica">ssp. lusitanica</string>
    <string name="SspFairmairei">ssp. fairmairei</string>
    <string name="FrequenciaMostreigExigua">* A causa de la baixa freqüència de mostreig utilitzada a la gravació o de la presència d\'un filtre de freqüències aplicat pel propi enregistrador, no s\'ha pogut comprovar si l\'espectre freqüencial del cant coincideix amb el d\'aquesta espècie. El nivell de semblança mostrat no té en compte aquest factor.</string>
    <string name="AdvertenciaCicadaBarbaraLusitanica">** De vegades, la superposició dels cants de diversos individus de <i>Cicada orni</i> pot fer semblar que el cant enregistrat és continu, com el de la <i>Cicada barbara lusitanica</i>.</string>
    <string name="Analitzant">Analitzant el cant</string>
    <string name="title_activity_explorador_fitxers">Selecciona un fitxer</string>
    <string name="DirectoriActual">Directori actual</string>
    <string name="Fitxer">fitxer</string>
    <string name="Fitxers">fitxers</string>
    <string name="FitxerNoValid">Fitxer no vàlid</string>
    <string name="SHaDeSeleccionarUnFitxerWav">s\'ha de seleccionar un fitxer .wav</string>
    <string name="Versio1.0">1.0</string>
    <string name="Copyright">Copyright &amp;#169; 2016 \t David Funosas Planas</string>
    <string name="DescripcioAplicacio">CicadaApp és una aplicació per a la identificació automàtica de cigales a partir d\'una gravació del seu cant. Les espècies incloses són totes aquelles amb presència real o potencial a Catalunya l\'any 2016.</string>
    <string name="DescripcioAplicacio2">La principal utilitat de l\'aplicació és la identificació de cigales del gènere <i>Tibicina</i>, pràcticament indistingibles auditivament i per a les quals els resultats de l\'anàlisi són altament fiables.</string>
    <string name="LlicenciaPart1">Aquesta aplicació és programari lliure; la podeu redistribuir i/o modificar sota els termes de la Llicència Pública General GNU que estableix la Free Software Foundation; bé la versió 2 de la llicència, bé qualsevol versió ulterior.</string>
    <string name="LlicenciaPart2">CicadaApp és distribuïda amb l\'esperança que sigui útil, però sense cap garantia; sense ni tan sols la garantia implícita de comercialització o adequació a un propòsit particular. Llegiu la GNU General Public License per a més detalls.</string>
    <string name="DAcord">D\'acord</string>
    <string name="Agraiments">Agraïments: A Thomas Hertach, Tomi Trilar i Pere Pons pel proveïment de gravacions de cants de cigala, i a Gerard Funosas pel disseny del logotip de l\'aplicació, en tots els casos de forma altruista i desinteressada</string>
    <string name="CorreuContacte">Correu de contacte per a qualsevol dubte o consulta sobre l\'aplicació: davidfunosas@gmail.com</string>
    <string name="MemoriaInsuficient">La memòria cau actualment disponible al dispositiu és insuficient per analitzar el cant enregistrat. Per evitar aquest problema es recomana que les gravacions siguin més curtes.</string>
    <string name="ErrorAnalisi">Error en analitzar la gravació</string>
    <string name="SenseCoincidencies">No s\'ha trobat coincidències.</string>
    <string name="CantFluix">El cant enregistrat és massa tènue per poder ser analitzat. S\'hauria de gravar l\'individu cantor de més a prop.</string>
    <string name="CicadaOrniDeFons">Només s\'ha pogut detectar el que probablement és una o diverses <i>Cicada orni</i> cantant de fons.</string>
    <string name="InstruccionsDUs">Instruccions d\'ús</string>
    <string name="InstruccionsIConsells">Instruccions i consells que cal tenir en compte a l\'hora d\'utilitzar l\'aplicació:</string>
    <string name="Instruccio1">Per enregistrar un cant de cigala tan sols cal fer clic primer a \"Inicia una gravació\" i posteriorment a la icona del micròfon per començar a gravar. El mòbil enregistrarà el cant fins que es cliqui el micròfon de nou. Els enregistraments quedaran desats a la carpeta cicadaAppRecordings amb un nom que indiqui el dia i l\'hora de la gravació.</string>
    <string name="Instruccio2">A l\'hora de fer les gravacions, és recomanable apuntar a l\'individu cantor amb la part del mòbil des d\'on es capta el so i fer-ho des de poca distància (no més de 5 metres), ja que els enregistradors dels mòbils estan pensats per gravar des d\'una separació d\'uns pocs centímetres respecte a la font del so i solen presentar dificultats a l\'hora de gravar sons llunyans. Prèviament a l\'anàlisi, les gravacions són amplificades tant com sigui possible sense distorsionar el so; tanmateix, una major distància respecte a l\'individu implicarà una pitjor qualitat de l\'enregistrament i, consegüentment, una menor fiabilitat dels resultats.</string>
    <string name="Instruccio3">En cas que hi hagi més d\'un individu cantant alhora en el moment de la gravació, ja siguin d\'espècies diferents o de la mateixa, n\'hi ha d\'haver un la potència del cant del qual destaqui clarament sobre la resta. Altrament, la superposició dels diversos cants podria distorsionar el resultat. L\'impacte causat per la presència d\'altres cants és especialment elevat en els casos de la <i>Tettigettalna argentata</i> i la <i>Cicadetta cerdaniensis</i>, atesa la tipologia característica del seu cant.</string>
    <string name="Instruccio4">Per evitar que els temps d\'anàlisi siguin elevats, s\'aconsella que la durada de les gravacions sigui moderada (d\'uns 5-10 segons).</string>
    <string name="Instruccio5">Per motius de prudència en la identificació, es recomana utilitzar els nivells de semblança que dóna l\'aplicació com a dada de suport o criteri addicional, i no confiar-hi cegament, especialment si en reproduir l\'enregistrament prèviament a l\'anàlisi comprovem que no se sent el cant de manera nítida.</string>
    <string name="LlistaEspeciesCurt">Llista d\'espècies</string>
    <string name="Fenologia">Quan es pot veure</string>
    <string name="Identificacio">Identificació</string>
    <string name="Habitat">Hàbitat</string>
    <string name="Cant">Cant</string>
    <string name="PerePons">Fotografia de Pere Pons</string>
    <string name="PacoFaluke">Fotografia de Paco Faluke</string>
    <string name="DanielRojas">Fotografia de Daniel Rojas</string>
    <string name="XavierDeYzaguirre">Fotografia de Xavier De Yzaguirre</string>
    <string name="MatijaGogala">Fotografia de Matija Gogala</string>
    <string name="ThomasHertach">Fotografia de Thomas Hertach</string>
    <string name="AntonioGarciaMaldonado">Fotografia d\'Antonio García Maldonado</string>
    <string name="CosminOvidiu">Fotografia de Cosmin Ovidiu</string>
    <string name="JeromeSueur">Fotografia de Jérome Sueur</string>
    <string name="StephanePuissant">Fotografia de Stéphane Puissant</string>
    <string name="VincentDerreumaux">Fotografia de Vincent Derreumaux</string>
    <string name="IgnasiJosepTejedor">Fotografia d\'Ignasi Josep Tejedor</string>
    <string name="JeanPierreLavigne">Fotografia de Jean-Pierre Lavigne</string>
    <string name="JoseManuelSesma">Fotografia de José Manuel Sesma</string>
    <string name="FerranTurmo">Fotografia de Ferran Turmo</string>
    <string name="MichelBoulard">Fotografia de Michel Boulard</string>
    <string name="FotografiaResultatsDanielRojas">*La fotografia mostrada a Resultats és de Daniel Rojas.</string>
    <string name="FotografiaResultatsCosminOvidiu">*La fotografia mostrada a Resultats és de Cosmin Ovidiu.</string>
    <string name="FotografiaResultatsVincentDerreumaux">*La fotografia mostrada a Resultats és de Vincent Derreumaux.</string>
    <string name="FotografiaResultatsIvanJesusTorresanoGarcia">*La fotografia mostrada a Resultats és d\'Iván Jesús Torresano García.</string>
    <string name="IdentificacioCicadaBarbaraLusitanica">Coloració gris-verdosa o gris-marronosa, amb una desena de taques fosques a cada ala. No presenta cap tret fisiològic que permeti distingir-la de manera fiable de la <i>Cicada orni</i>: només poden ser diferenciades pel seu cant.</string>
    <string name="HabitatCicadaBarbaraLusitanica">El seu hàbitat principal són les zones amb vegetació arbòria oberta, especialment oliverars, garroferars o pinedes, tot i que també pot trobar-se en zones amb vegetació arbustiva elevada.</string>
    <string name="CantCicadaBarbaraLusitanica">Pot cantar des d\'un tronc o des d\'una branca gruixuda. Quan un mascle comença a cantar, altres mascles propers poden seguir-lo i acabar formant un cor de cants simultanis.</string>
    <string name="IdentificacioCicadaOrni">Cos d\'uns 28 mm de longitud i coloració gris-verdosa o gris-marronosa amb una desena de taques fosques a cada ala, característica pròpia del gènere. No presenta cap tret fisiològic que permeti distingir-la de manera fiable de la <i>Cicada barbara</i>: només poden ser diferenciades pel seu cant.</string>
```

<string name="HabitatCicadaOrni">És l\'espècie més comuna a Catalunya. És habitual en una gran varietat de boscos i arbredes, amb especial abundància a les pinedes, però també present en màquies o garrigars, sempre que siguin prou obertes perquè la llum del sol hi pugui penetrar.</string>
<string name="CantCicadaOrni">Pot cantar des d\'un tronc o des d\'una branca gruixuda. És habitual sentir diversos mascles cantant alhora, formant un cor.</string>
<string name="IdentificacioCicadatraAtra">Cos d\'uns 18 mm de longitud i coloració negrosa. Té dues o tres taques negres a les ales anteriors.</string>
<string name="HabitatCicadatraAtra">Habita una gran varietat d\'ambients, des de màquies o garrigars fins a boscos caducifolis o de ribera, amb preferència per comunitats vegetals amb un estrat de vegetació elevada. També es pot trobar en parcs o jardins urbans.</string>
<string name="CantCicadatraAtra">Pot cantar des de troncs, branques o tiges, sempre mirant cap avall.</string>
<string name="IdentificacioCicadettaBrevipennis">Cos d\'uns 18 mm de longitud i coloració negra/marronosa, amb les vores dels segments de l\'abdomen de color taronja. No presenta cap tret fisiològic que permeti distingir-la de manera fiable de la Cicadetta cerdaniensis: només poden ser diferenciades pel seu cant.</string>
<string name="HabitatCicadettaBrevipennis">Pot habitar ambients molt diversos, des de planes fins a boscos, però es troba bàsicament a les zones obertes de mitjana i alta muntanya.</string>
<string name="CantCicadettaBrevipennis">Canta des de tiges o branquetes de pocs mil·límetres de diàmetre i des d\'alçades molt variables.</string>
<string name="IdentificacioCicadettaCerdaniensis">Cos d\'uns 18 mm de longitud i coloració negra/marronosa, amb les vores dels segments de l\'abdomen de color taronja. No presenta cap tret fisiològic que permeti distingir-la de manera fiable de la Cicadetta brevipennis: només poden ser diferenciades pel seu cant.</string>
<string name="HabitatCicadettaCerdaniensis">Es pot trobar a zones de mitjana i alta muntanya, habitualment en landes denses però també en vegetació arbòria. Tenen preferència per plantes espinoses com ara el roser silvestre o l\'esbarzer.</string>
<string name="CantCicadettaCerdaniensis">Canta des de tiges o branquetes de pocs mil·límetres de diàmetre i des d\'alçades molt variables.</string>
<string name="IdentificacioHilaphuraVaripes">Coloració negrosa, amb quatre taques de color crema, dues al tòrax i dues a l\'abdomen, que formen una creu negra entremig.</string>
<string name="HabitatHilaphuraVaripes">Habita espais oberts, tant de terra baixa com de muntanya mitjana, amb clima sec i càlid. Prefereix terres no conreades amb poca vegetació o amb un clar predomini de la vegetació herbàcia per sobre de la llenyosa.</string>
<string name="CantHilaphuraVaripes">Canta des de les tiges de les plantes herbàcies amb uns pocs mil·límetres de diàmetre.</string>
<string name="IdentificacioLyristesPlebejus">La més gran de les que es pot trobar a Catalunya, amb un cos d\'uns 35 mm de longitud. Presenta una coloració cendrosa, amb un “collar” de color taronja apagat (verdós quan surt de l\'exúvia) i sovint dues taques fosques a les ales anteriors.</string>
<string name="HabitatLyristesPlebejus">Habita principalment zones amb vegetació elevada i llenyosa, però també es pot trobar en ambients no boscosos, com ara màquies o garrigues poc denses.</string>
<string name="CantLyristesPlebejus">Sol cantar des de branques gruixudes i de manera individual, sense formar cors amb altres mascles.</string>
<string name="IdentificacioTettigettalnaArgentata">Cos d\'uns 17 mm de longitud i coloració gris-marronosa, amb el ventre clar i dues taques ocres a la part superior de l\'abdomen.</string>
<string name="HabitatTettigettalnaArgentata">Pot habitar una gran varietat de comunitats vegetals de terra baixa i muntanya mitjana, des de landes fins a boscos i arbredes. Abunda especialment en ambients secs i càlids amb presència de vegetació llenyosa elevada.</string>
<string name="CantTettigettalnaArgentata">Canta sobretot des de tiges de pocs mil·límetres de diàmetre, però també pot fer-ho des de branques o troncs d\'alçades molt diverses.</string>
<string name="IdentificacioTettigettulaPygmea">La més menuda de les que pot haver-hi a Catalunya, amb un cos d\'uns 13 mm de longitud. Presenta una coloració negra o gris molt fosc per tot el cos, inclosa la zona ventral (a diferència de la Tettigettalna argentata). També té dues taques de color ocre a la part superior de l\'abdomen.</string>
<string name="HabitatTettigettulaPygmea">Habita boscos poc densos, que permetin la penetració de la llum del sol, i zones amb vegetació arbustiva elevada, tant de terres baixes com de muntanya mitjana. Prefereix els ambients secs i càlids.</string>
<string name="CantTettigettulaPygmea">Canta des de branquetes, branques o troncs amb diàmetres que van des d\'uns pocs mil·límetres a desenes de centímetres, i ho fan principalment des d\'alçades superiors als dos metres.</string>
<string name="IdentificacioTibicinaCorsicaFairmairei">Cos d\'uns 24 mm de longitud, de colors gris fosc i taronja i amb els primers segments de l\'abdomen completament grisos. Igual que la Tibicina tomentosa, té els nervis alars de color blanc groguenc a la base i negres a la punta.</string>
<string name="HabitatTibicinaCorsicaFairmairei">Habita ambients esteparis de zones baixes, així com garrigars poc densos.</string>
<string name="CantTibicinaCorsicaFairmairei">Canta des de tiges o branquetes amb un diàmetre d\'entre mig i un centímetre, a menys de dos metres d\'alçada i des de llocs assolellats.</string>
<string name="IdentificacioTibicinaGarricola">Cos d\'uns 26 mm de longitud, de coloració negra i taronja. Presenta quatre taques taronges a la part superior del tòrax, com la Tibicina quadrisignata, i nervis alars de dues coloracions: blanc groguenc o verdós i negre.</string>
<string name="HabitatTibicinaGarricola">Habita ambients secs i arbustius de zones baixes, amb especial predilecció pel garric i l\'estepa, però també es pot trobar en zones amb vegetació arbòria oberta.</string>
<string name="CantTibicinaGarricola">Canta des de l\'interior dels arbustos, sense que li toqui el sol i en suports d\'entre mig i uns pocs centímetres de diàmetre.</string>
<string name="IdentificacioTibicinaHaematodes">Cos d\'uns 30 mm de longitud, amb una coloració dominant negra i taronja. Els nervis alars són completament taronges, excepte en la forma viridinervis, poc habitual, en què aquests són de color verd i la coloració del cos és lleugerament més apagada.</string>
<string name="HabitatTibicinaHaematodes">Habita zones de vegetació arbòria o vegetació arbustiva elevada, sobretot en terres baixes amb clima sec i càlid. Abunda especialment en alzinars, rouredes i plantacions de vinya, i també es pot trobar a parcs i jardins urbans.</string>
<string name="CantTibicinaHaematodes">Canta des de branques amb diversos centímetres de diàmetre i a una altitud elevada, sovint a prop de la capçada dels arbres.</string>
<string name="IdentificacioTibicinaQuadrisignata">Cos d\'uns 27 mm de longitud i de color negre o gris molt fosc, amb algunes parts taronges. Els nervis alars són foscos i presenta, igual que la Tibicina garricola, quatre taques taronges al tòrax.</string>
<string name="HabitatTibicinaQuadrisignata">Habita boscos oberts i zones amb vegetació arbustiva, tant de terres baixes com de muntanya mitjana. Es pot trobar en alzinars, pinedes, suredes, oliverars o castanyedes, entre d\'altres, però també en màquies o landes.</string>
<string name="CantTibicinaQuadrisignata">Canta des de branques amb diversos centímetres de diàmetre i a una alçada superior als dos metres, sovint a prop de la capçada dels arbres.</string>
<string name="IdentificacioTibicinaTomentosa">Cos d\'uns 26 mm de longitud, de color negre o gris molt fosc i amb les vores dels segments de l\'abdomen i les potes de color taronja. Els nervis alars són blancs o d\'un to groguenc a la base i negres a la punta de les ales.</string>
<string name="HabitatTibicinaTomentosa">Habita ambients oberts especialment secs i càlids, amb poca o nul·la presència de vegetació arbòria.</string>
<string name="CantTibicinaTomentosa">Canta des de les tiges de les plantes herbàcies amb uns pocs mil·límetres de diàmetre.</string>
</resources>

strings.xml (en):

<resources>
<string name="app_name">CicadaApp</string>

<string name="LesEspecies">The species</string>
<string name="IniciaUnaGravacio">Start a recording</string>
<string name="AnalitzaUnFitxer">Analyze a file</string>
<string name="title_activity_llista_especies">LlistaEspecies</string>
<string name="title_activity_enregistrador_audio">Recording</string>
<string name="title_activity_resultats">Results</string>
<string name="title_activity_fitxa_especie">Species information</string>
<string name="LlistaEspecies">List of the cicada species with real or potential presence in Catalonia</string>
<string name="Catala">Català</string>
<string name="Castella">Castellano</string>
<string name="Angles">English</string>
<string name="Frances">French</string>
<string name="QuantA">About CicadaApp</string>
<string name="Instruccions">Use instructions</string>
<string name="Reprodueix">Play</string>
<string name="Analitza">Analyze</string>
<string name="Descarta">Discard</string>
<string name="Atura">Stop</string>
<string name="Resultats">Results</string>
<string name="ReproduccioEnCurs">Playing</string>
<string name="SemblancesTrobades">Similarities found</string>
<string name="NoHiHaResultats">No coincidences have been found</string>
<string name="CicadaOrni">Cicada orni</string>
<string name="CicadaBarbaraLusitanica">Cicada barbara lusitanica</string>
<string name="CicadatraAtra">Cicadatra atra</string>
<string name="CicadettaBrevipennis">Cicadetta brevipennis</string>
<string name="CicadettaCerdaniensis">Cicadetta cerdaniensis</string>
<string name="HilaphuraVaripes">Hilaphura varipes</string>
<string name="LyristesPlebejus">Lyristes plebejus</string>
<string name="TettigettalnaArgentata">Tettigettalna argentata</string>
<string name="TettigettulaPygmea">Tettigettula pygmea</string>
<string name="TibicinaCorsicaFairmairei">Tibicina corsica fairmairei</string>
<string name="TibicinaGarricola">Tibicina garricola</string>
<string name="TibicinaHaematodes">Tibicina haematodes</string>
<string name="TibicinaQuadrisignata">Tibicina quadrisignata</string>
<string name="TibicinaTomentosa">Tibicina tomentosa</string>
<string name="CicadaBarbara">Cicada barbara</string>
<string name="TibicinaCorsica">Tibicina corsica</string>
<string name="SspLusitanica">ssp. lusitanica</string>
<string name="SspFairmairei">ssp. fairmairei</string>
<string name="FrequenciaMostreigExigua">* Because of the low sampling frequency used or the presence of a frequency filter applied by the recorder, it has not been possible to check if the frequency spectrum of the song coincides with that of this species. The shown similarity coefficient does not take into account this factor.</string>
<string name="AdvertenciaCicadaBarbaraLusitanica">** Sometimes, the overlapping of the songs of several individuals of Cicada orni can give the impression that the recorded song is continuous, like the one of Cicada barbara lusitanica.</string>
<string name="Analitzant">Analyzing the song</string>

<string name="title_activity_explorador_fitxers">Select a file</string>
<string name="DirectoriActual">Current directory</string>
<string name="Fitxer">file</string>
<string name="Fitxers">files</string>
<string name="FitxerNoValid">Invalid file</string>
<string name="SHaDeSeleccionarUnFitxerWav">the selected file must have .wav format</string>
<string name="Versio1.0">1.0</string>
<string name="Copyright">Copyright © 2016 \t David Funosas Planas</string>
<string name="DescripcioAplicacio">CicadaApp is an application for the automatic identification of cicadas from a recording of their song. The species included are all those with actual or potential presence in Catalonia in 2016.</string>
<string name="DescripcioAplicacio2">The main utility of the application is the identification of cicadas from the genus Tibicina, virtually indistinguishable aurally and for which the analysis\' results are highly reliable.</string>
<string name="LlicenciaPart1">This application is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.</string>
<string name="LlicenciaPart2">CicadaApp is distributed in the hope that it will be useful, but without any warranty; without even the implied warranty of merchantability or fitness for a particular purpose. See the GNU General Public License for more details.</string>
<string name="DAcord">OK</string>
<string name="Agraiments">Acknowledgements: To Thomas Hertach, Tomi Trilar and Pere Pons for the provision of cicada song recordings, and to Gerard Funosas for the design of the application\'s logotype, in all cases in an altruistic and disinterested way</string>
<string name="CorreuContacte">Contact mail for any doubt or question about the application: davidfunosas@gmail.com</string>
<string name="MemoriaInsuficient">The cache memory currently available in the device is insufficient to analyse the recorded song. To avoid this problem, it is recommended that the recordings be shorter.</string>
<string name="ErrorAnalisi">Error when analysing the recording</string>
<string name="SenseCoincidencies">Coincidencies have not been found for the recorded song.</string>
<string name="CantFluix">The recorded song is too faint to be analysed. The individual should be recorded closer up.</string>
<string name="CicadaOrniDeFons">It has only been detected what is probably one or more Cicada orni singing in the background.</string>
<string name="InstruccionsDUs">Instructions for use</string>
<string name="InstruccionsIConsells">Instructions and advices that have to be taken into account when using the application:</string>
<string name="Instruccio1">To record a cicada song you just need to click \"Start a recording\" and subsequently the microphone icon in order to start. The phone will be recording the song until you click the icon again. The recordings will be stored in the folder cicadaAppRecordings with a name that indicates the date and the time of the recording.</string>
<string name="Instruccio2">When recording, it is advisable to point to the singing individual with the part of the phone from where the sound is received and to do it from a close distance (not more than 5 metres), since the phone recorders are mainly designed to record from just a few centimetres away from the sound source and they often have difficulties when recording distant sounds. Prior to the analysis, the recordings are amplified as much as possible without distorting the sound; however, a greater distance from the individual will imply a poorer quality of the recording and, consequently, a lower reliability of the results.</string>
<string name="Instruccio3">If there are several individuals singing at the same time during the recording, whether they belong to different species or the same, there has to be an individual whose song volume stands out clearly above the rest. Otherwise, the superposition of the multiple songs could distort the result. The impact caused by the presence of other songs is particularly high in the case of Tettigettalna argentata and Cicadetta cerdaniensis, given the characteristic typology of their song.</string>
<string name="Instruccio4">To avoid long runtimes, a moderate recording duration (about 5-10 seconds) is recommended.</string>
<string name="Instruccio5">For reasons of caution in the identification, it is advised to use the similarity levels given by the application as supporting data or additional criterion, and not to believe them blindly, especially if when playing the recording prior to the analysis we find that the song can not be heard clearly.</string>
<string name="LlistaEspeciesCurt">Species list</string>
<string name="Fenologia">When it can be seen</string>
<string name="Identificacio">Identification</string>
<string name="Habitat">Habitat</string>
<string name="Cant">Song</string>
<string name="PerePons">Photo by Pere Pons</string>
<string name="PacoFaluke">Photo by Paco Faluke</string>
<string name="DanielRojas">Photo by Daniel Rojas</string>
<string name="XavierDeYzaguirre">Photo by Xavier De Yzaguirre</string>
<string name="MatijaGogala">Photo by Matija Gogala</string>
<string name="ThomasHertach">Photo by Thomas Hertach</string>
<string name="AntonioGarciaMaldonado">Photo by Antonio García Maldonado</string>
<string name="CosminOvidiu">Photo by Cosmin Ovidiu</string>
<string name="JeromeSueur">Photo by Jérome Sueur</string>
<string name="StephanePuissant">Photo by Stéphane Puissant</string>
<string name="VincentDerreumaux">Photo by Vincent Derreumaux</string>
<string name="IgnasiJosepTejedor">Photo by Ignasi Josep Tejedor</string>
<string name="JeanPierreLavigne">Photo by Jean-Pierre Lavigne</string>
<string name="JoseManuelSesma">Photo by José Manuel Sesma</string>
<string name="FerranTurmo">Photo by Ferran Turmo</string>
<string name="MichelBoulard">Photo by Michel Boulard</string>
<string name="FotografiaResultatsDanielRojas">*The photo shown in Results is by Daniel Rojas.</string>
<string name="FotografiaResultatsCosminOvidiu">*The photo shown in Results is by Cosmin Ovidiu.</string>
<string name="FotografiaResultatsVincentDerreumaux">*The photo shown in Results is by Vincent Derreumaux.</string>
<string name="FotografiaResultatsIvanJesusTorresanoGarcia">*The photo shown in Results is by Iván Jesús Torresano García.</string>
<string name="IdentificacioCicadaBarbaraLusitanica">Greenish or brownish grey coloured body, with around ten dark spots in each wing. It does not have any physiological trait that allows to reliably distinguish it from Cicada orni: they can only be differentiated by their song.</string>
<string name="HabitatCicadaBarbaraLusitanica">Its main habitat are areas with open tree vegetation, especially olive or carob tree groves or pine forests, though it can also be found in areas with high shrubbery.</string>
<string name="CantCicadaBarbaraLusitanica">It can sing from a trunk or from a thick branch. When a male starts singing, other nearby males can follow it and end up forming a choir of simultaneous songs.</string>
<string name="IdentificacioCicadaOrni">Body length of about 28 mm, with greenish or brownish grey colour throughout the body and around ten dark spots in each wing. It does not have any physiological trait that allows to reliably distinguish it from Cicada barbara: they can only be differentiated by their song.</string>
<string name="HabitatCicadaOrni">It is the most common species in Catalonia. It inhabits a great variety of woods and forests, with particular abundance in pine forests, but it is also present in maquis shrublands or garrigues, provided they are open enough to let sunlight penetrate.</string>
<string name="CantCicadaOrni">It can sing from a trunk or from a thick branch. It is common to hear several males singing at the same time, forming a choir.</string>
<string name="IdentificacioCicadatraAtra">Body length of about 18 mm, with blackish coloration and two or three black spots in the forewings.</string>
<string name="HabitatCicadatraAtra">It inhabits a great variety of environments, from maquis shrublands or garrigues to deciduous or riparian forests, with preference for plant communities with a layer of high vegetation. It can also be found in urban parks or gardens.</string>
<string name="CantCicadatraAtra">It can sing from trunks, branches or stems, always looking down.</string>
<string name="IdentificacioCicadettaBrevipennis">Body length of about 18 mm, with blackish/brownish coloration and the edges of the abdomen segments of an orange colour. It does not have any physiological trait that allows to reliably distinguish it from Cicadetta cerdaniensis: they can only be differentiated by their song.</string>
<string name="HabitatCicadettaBrevipennis">It can inhabit very diverse environments, from plains to forests, but we can basically find it in medium and high altitude open areas.</string>
<string name="CantCicadettaBrevipennis">It sings from stems or branches with a diameter of just a few millimetres and from widely varying heights.</string>
<string name="IdentificacioCicadettaCerdaniensis">Body length of about 18 mm, with blackish/brownish coloration and the edges of the abdomen segments of an orange colour. It does not have any physiological trait that allows to reliably distinguish it from Cicadetta brevipennis : they can only be differentiated by their song.</string>
<string name="HabitatCicadettaCerdaniensis">It can be found in areas of medium and high altitude, usually in dense heathlands but also in forest areas. They have preference for thorny plants like dog-roses or wild blackberrys.</string>
<string name="CantCicadettaCerdaniensis">It sings from stems or branches with a diameter of just a few millimetres and from widely varying heights.</string>
<string name="IdentificacioHilaphuraVaripes">Blackish colour, with for cream-coloured spots, two on the thorax and two on the abdomen, forming a black cross in the middle.</string>
<string name="HabitatHilaphuraVaripes">It inhabits open spaces, in both lowlands and mid-mountain, with dry and warm weather. It prefers uncultivated lands with little vegetation or areas with a clear predominance of herbaceous vegetation above the woody.</string>
<string name="CantHilaphuraVaripes">It sings from the stems of herbaceous plants with a diameter of just a few millimetres.</string>
<string name="IdentificacioLyristesPlebejus">The largest among the ones that can be found in Catalonia, with a body length of about 35 mm. It has an ashy colour, with a dull orange (greenish when it leaves the exuvia) “collar” and often two dark spots in the forewings.</string>
<string name="HabitatLyristesPlebejus">It mainly inhabits areas with high and woody vegetation, but it can also be found in non-forest environments, such as open garrigues or maquis.</string>
<string name="CantLyristesPlebejus">It usually sings from thick branches and individually, without forming choirs with other males.</string>
<string name="IdentificacioTettigettalnaArgentata">Body about 17 mm long with a brownish grey colour, a light belly and two ochre spots on the upper abdomen.</string>
<string name="HabitatTettigettalnaArgentata">It can inhabit a great variety of plant communities in lowlands and in mid-mountain, from moorlands to forests and woods. It abounds especially in hot, dry environments with high woody vegetation.</string>
<string name="CantTettigettalnaArgentata">It sings mainly from stems with a diameter of just a few millimetres, but it can also do it from branches or trunks from very different heights.</string>
<string name="IdentificacioTettigettulaPygmea">The smallest among the ones that can be found in Catalonia, with a body length of about 13 mm. The dominant colour throughout its body, including the ventral area (unlike Tettigettalna argentata), is black or very dark grey. It also has two ochre spots in the upper abdomen.</string>
<string name="HabitatTettigettulaPygmea">It inhabits open forests that allow the penetration of sunlight and areas with high shrubbery, both in lowlands and in mid-mountain. It prefers dry and warm environments.</string>

<string name="CantTettigettulaPygmea">It sings from stems, branches or trunks with diameters ranging from a few millimetres to tens of centimetres, and they do it mainly from over 2 m altitudes.</string>
<string name="IdentificacioTibicinaCorsicaFairmairei">Body length of about 24 mm, with dark grey and orange as dominant colours of the body and the first abdomen segments completely grey. Like Tibicina tomentosa, its alar ribs are yellowish-white at the base and black at the tip.</string>
<string name="HabitatTibicinaCorsicaFairmairei">It inhabits lowland steppe environments and low density garrigues.</string>
<string name="CantTibicinaCorsicaFairmairei">It sings from stems or branches with a diameter of between a half and one centimetre, mainly from sunny places under 2 m altitudes.</string>
<string name="IdentificacioTibicinaGarricola">Body about 26 mm long, with black and orange colours. It has four orange spots on the thorax, like Tibicina quadrisignata, and alar ribs of two colours: black and yellowish or greenish white.</string>
<string name="HabitatTibicinaGarricola">It inhabits dry shrublands in lowlands, with special predilection for kermes oaks and rockroses, but it can also be found in areas with open tree vegetation.</string>
<string name="CantTibicinaGarricola">It sings from inside the bushes, without the sunlight touching it and in supports with a diameter of between a half and some centimetres.</string>
<string name="IdentificacioTibicinaHaematodes">Body about 30 mm long, with black and orange as dominant colours. The alar ribs are completely orange, except in the viridinervis form, quite unusual, in which case these ribs are green and the body colour is slightly duller.</string>
<string name="HabitatTibicinaHaematodes">It lives in areas with tree or high shrub vegetation, mainly in lowlands with dry and warm weather. It abounds especially in holm and downy oak woods and in vineyard plantations, and it can also be found in urban parks and gardens.</string>
<string name="CantTibicinaHaematodes">It sings from branches with several centimetres of diameter and at a high altitude, often from near the treetops.</string>
<string name="IdentificacioTibicinaQuadrisignata">Body about 27 mm long and with a very dark grey or black colour, except for some orange-coloured parts. The alar ribs are dark and it has, like Tibicina garricola, four orange spots on the thorax.</string>
<string name="HabitatTibicinaQuadrisignata">It inhabits open forests and areas with shrubbery vegetation, both in lowlands and in mid-mountain. It can be found in holm oak, cork oak or chestnut woods, pine forests or olive groves, among others, but also in maquis or heaths.</string>
<string name="CantTibicinaQuadrisignata">It sings from branches with a diameter of several centimetres and from over 2 m altitudes, often near the treetops.</string>
<string name="IdentificacioTibicinaTomentosa">Body about 26 mm long and with a very dark grey or black colour, except for the orange-coloured legs and edges of the abdomen segments. The alar ribs are white or yellowish in the base and black in the wingtips.</string>
<string name="HabitatTibicinaTomentosa">It inhabits especially dry and warm open environments, with little or no presence of tree vegetation.</string>
<string name="CantTibicinaTomentosa">It sings from the stems of herbaceous plants with a diameter of just a few millimetres.</string>
</resources>

strings.xml (es):

<resources>
<string name="app_name">CicadaApp</string>

<string name="LesEspecies">Las especies</string>
<string name="IniciaUnaGravacio">Iniciar una grabación</string>
<string name="AnalitzaUnFitxer">Analizar un archivo</string>
<string name="title_activity_llista_especies">LlistaEspecies</string>
<string name="title_activity_enregistrador_audio">Grabación en curso</string>
<string name="title_activity_resultats">Resultados</string>
<string name="title_activity_fitxa_especie">Ficha de la especie</string>
<string name="LlistaEspecies">Lista de las especies de cigarra con presencia real o potencial en Cataluña</string>
<string name="Catala">Català</string>
<string name="Castella">Castellano</string>
<string name="Angles">English</string>
<string name="Frances">French</string>
<string name="QuantA">En cuanto a CicadaApp</string>
<string name="Instruccions">Instrucciones de uso</string>
<string name="Reprodueix">Reproducir</string>
<string name="Analitza">Analizar</string>
<string name="Descarta">Descartar</string>
<string name="Atura">Detener</string>
<string name="Resultats">Resultados</string>
<string name="ReproduccioEnCurs">Reproducción en curso</string>
<string name="SemblancesTrobades">Similitudes encontradas</string>
<string name="NoHiHaResultats">No se ha encontrado coincidencias</string>
<string name="CicadaOrni">Cicada orni</string>
<string name="CicadaBarbaraLusitanica">Cicada barbara lusitanica</string>
<string name="CicadatraAtra">Cicadatra atra</string>
<string name="CicadettaBrevipennis">Cicadetta brevipennis</string>
<string name="CicadettaCerdaniensis">Cicadetta cerdaniensis</string>
<string name="HilaphuraVaripes">Hilaphura varipes</string>
<string name="LyristesPlebejus">Lyristes plebejus</string>
<string name="TettigettalnaArgentata">Tettigettalna argentata</string>
<string name="TettigettulaPygmea">Tettigettula pygmea</string>
<string name="TibicinaCorsicaFairmairei">Tibicina corsica fairmairei</string>
<string name="TibicinaGarricola">Tibicina garricola</string>
<string name="TibicinaHaematodes">Tibicina haematodes</string>
<string name="TibicinaQuadrisignata">Tibicina quadrisignata</string>
<string name="TibicinaTomentosa">Tibicina tomentosa</string>
<string name="CicadaBarbara">Cicada barbara</string>
<string name="TibicinaCorsica">Tibicina corsica</string>
<string name="SspLusitanica">ssp. lusitanica</string>
<string name="SspFairmairei">ssp. fairmairei</string>
<string name="FrequenciaMostreigExigua">* A causa de la baja frecuencia de muestreo utilizada en la gravación o de la presencia de un filtro de frecuencias aplicado por el propio enregistrador, no se ha podido comprovar si el espectro frecuencial del canto coincide con el de esta especie. El nivel de similitud mostrado no tiene en cuenta este factor.</string>
<string name="AdvertenciaCicadaBarbaraLusitanica">** A veces, la superposición de los cantos de varios individuos de Cicada orni puede hacer parecer que el canto grabado es continuo, como el de la Cicada barbara lusitanica.</string>
<string name="Analitzant">Analizando el canto</string>
<string name="title_activity_explorador_fitxers">Selecciona un archivo</string>
<string name="DirectoriActual">Directorio actual</string>
<string name="Fitxer">archivo</string>
<string name="Fitxers">archivos</string>
<string name="FitxerNoValid">Archivo no válido</string>
<string name="SHaDeSeleccionarUnFitxerWav">se debe seleccionar un archivo .wav</string>
<string name="Versio1.0">1.0</string>
<string name="Copyright">Copyright © 2016 \t David Funosas Planas</string>
<string name="DescripcioAplicacio">CicadaApp es una aplicación para la identificación automática de cigarras a partir de una grabación de su canto. Las especies incluidas son todas aquellas con presencia real o potencial en Cataluña en el año 2016.</string>
<string name="DescripcioAplicacio2">La principal utilidad de la aplicación es la identificación de cigarras del género Tibicina, prácticamente indistinguibles auditivamente y para las cuales los resultados del análisis son altamente fiables.</string>
<string name="LlicenciaPart1">Esta aplicación es software libre; la podéis redistribuir y/o modificar bajo los términos de la Licencia Pública General GNU que establece la Free Software Foundation; bien la versión 2 de la licencia, bien cualquier versión ulterior.</string>
<string name="LlicenciaPart2">CicadaApp es distribuida con la esperanza de que sea útil, pero sin ninguna garantía; ni siquiera la garantía implícita de comercialización o adecuación a un propósito particular. Leed la GNU General Public License para más detalles.</string>
<string name="DAcord">Aceptar</string>
<string name="Agraiments">Agradecimientos: A Thomas Hertach, Tomi Trilar y Pere Pons por la provisión de grabaciones de cantos de cigarra, y a Gerard Funosas por el diseño del logotipo de la aplicación, en todos los casos de forma altruista y desinteresada</string>
<string name="CorreuContacte">Correo de contacto para cualquier duda o consulta sobre la aplicación: davidfunosas@gmail.com</string>
<string name="MemoriaInsuficient">La memoria caché actualmente disponible en el dispositivo es insuficiente para analizar el canto grabado. Para evitar este problema se recomienda que las grabaciones sean más cortas.</string>
<string name="ErrorAnalisi">Error al analizar la grabación</string>
<string name="SenseCoincidencies">No se ha encontrado coincidencias.</string>
<string name="CantFluix">El canto grabado es demasiado tenue para poder ser analizado. Se debería grabar el individuo cantor más de cerca.</string>
<string name="CicadaOrniDeFons">Sólo se ha podido detectar lo que probablemente es una o varias Cicada orni cantando de fondo.</string>
<string name="InstruccionsDUs">Instrucciones de uso</string>
<string name="InstruccionsIConsells">Instrucciones y consejos que se debe tener en cuenta a la hora de usar la aplicación:</string>
<string name="Instruccio1">Para grabar un canto de cigarra tan sólo hace falta hacer clic primero a \"Iniciar una grabación\" y posteriormente al icono del micrófono para empezar a grabar. El móvil grabará el canto hasta que se clique el micrófono de nuevo. Las grabaciones quedarán guardadas en la carpeta cicadaAppRecordings con un nombre que indique el día y la hora de la grabación.</string>
<string name="Instruccio2">A la hora de hacer las grabaciones, es recomendable apuntar al individuo cantor con la parte del móvil desde la que se capta el sonido y hacerlo desde poca distancia (no más de 5 metros), ya que los grabadores de los móviles están pensados para grabar desde una separación de unos pocos centímetros respecto a la fuente de sonido y suelen presentar dificultades a la hora de grabar sonidos lejanos. Previamente al análisis, las grabaciones son amplificadas tanto como sea posible sin distorsionar el sonido; sin embargo, una mayor distancia respecto al individuo implicará una peor calidad de la grabación y, consiguientemente, una menor fiabilidad de los resultados.</string>
<string name="Instruccio3">En caso de que haya más de un individuo cantando al mismo tiempo en el momento de la grabación, ya sean de especies distintas o de la misma, tiene que haber uno la potencia de cuyo canto destaque claramente sobre el resto. Si no, la superposición de los múltiples cantos podría distorsionar el resultado. El impacto causado por la presencia de otros cantos es especialmente elevado en los casos de la Tettigettalna argentata y la Cicadetta cerdaniensis, dada la tipología característica de su canto.</string>
<string name="Instruccio4">Para evitar que los tiempos de análisis sean elevados, se aconseja que la duración de las grabaciones sea moderada (de unos 5-10 segundos).</string>
<string name="Instruccio5">Por motivos de prudencia en la identificación, se recomienda utilizar los niveles de similitud que da la aplicación como dato de apoyo o criterio adicional, y no confiar en ellos ciegamente, especialmente si al reproducir la

grabación previamente al análisis comprobamos que no se oye el canto de forma nítida.</string>
<string name="LlistaEspeciesCurt">Lista de especies</string>
<string name="Fenologia">Cuándo se puede ver</string>
<string name="Identificacio">Identificación</string>
<string name="Habitat">Hábitat</string>
<string name="Cant">Canto</string>
<string name="PerePons">Fotografía de Pere Pons</string>
<string name="PacoFaluke">Fotografía de Paco Faluke</string>
<string name="DanielRojas">Fotografía de Daniel Rojas</string>
<string name="XavierDeYzaguirre">Fotografía de Xavier De Yzaguirre</string>
<string name="MatijaGogala">Fotografía de Matija Gogala</string>
<string name="ThomasHertach">Fotografía de Thomas Hertach</string>
<string name="AntonioGarciaMaldonado">Fotografía de Antonio García Maldonado</string>
<string name="CosminOvidiu">Fotografía de Cosmin Ovidiu</string>
<string name="JeromeSueur">Fotografía de Jérome Sueur</string>
<string name="StephanePuissant">Fotografía de Stéphane Puissant</string>
<string name="VincentDerreumaux">Fotografía de Vincent Derreumaux</string>
<string name="IgnasiJosepTejedor">Fotografía de Ignasi Josep Tejedor</string>
<string name="JeanPierreLavigne">Fotografía de Jean-Pierre Lavigne</string>
<string name="JoseManuelSesma">Fotografía de José Manuel Sesma</string>
<string name="FerranTurmo">Fotografía de Ferran Turmo</string>
<string name="MichelBoulard">Fotografía de Michel Boulard</string>
<string name="FotografiaResultatsDanielRojas">*La fotografía mostrada en Resultados es de Daniel Rojas.</string>
<string name="FotografiaResultatsCosminOvidiu">*La fotografía mostrada en Resultados es de Cosmin Ovidiu.</string>
<string name="FotografiaResultatsVincentDerreumaux">*La fotografía mostrada en Resultados es de Vincent Derreumaux.</string>
<string name="FotografiaResultatsIvanJesusTorresanoGarcia">*La fotografía mostrada en Resultats es de Iván Jesús Torresano García.</string>
<string name="IdentificacioCicadaBarbaraLusitanica">Coloración gris-verdosa o gris-parduzca con una decena de manchas oscuras en cada ala, característica propia del género. No presenta ningún rasgo fisiológico que permita distinguirla de forma fiable de la Cicada orni: sólo pueden ser diferenciadas por su canto.</string>
<string name="HabitatCicadaBarbaraLusitanica">Su hábitat principal son las zonas con vegetación arbórea abierta, especialmente olivares, garrofales y pinedas, aunque también puede encontrarse en zonas con vegetación arbustiva elevada.</string>
<string name="CantCicadaBarbaraLusitanica">Puede cantar desde un tronco o desde una rama gruesa. Cuando un macho empieza a cantar, otros machos cercanos pueden seguirlo y acabar formando un coro de cantos simultáneos.</string>
<string name="IdentificacioCicadaOrni">Cuerpo de unos 28 mm de longitud y coloración gris-verdosa o gris-parduzca, con una decena de manchas oscuras en cada ala. No presenta ningún rasgo fisiológico que permita distinguirla de forma fiable de la Cicada barbara: sólo pueden ser diferenciadas por su canto.</string>
<string name="HabitatCicadaOrni">Es la especie más común en Cataluña. Es habitual en una gran variedad de bosques y arboledas, con especial abundancia en las pinedas, pero también presente en maquias o carrascales, siempre que sean lo suficientemente abiertos para que la luz del sol pueda penetrar.</string>
<string name="CantCicadaOrni">Puede cantar desde un tronco o desde una rama gruesa. Es habitual oír varios machos cantando al mismo tiempo, formando un coro.</string>
<string name="IdentificacioCicadatraAtra">Cuerpo de unos 18 mm de longitud y coloración negruzca. Tiene dos o tres manchas negras en las alas anteriores.</string>
<string name="HabitatCicadatraAtra">Habita una gran variedad de ambientes, desde maquias o carrascales hasta bosques caducifolios o de ribera, con preferencia por comunidades vegetales con un estrato de vegetación elevada. También se puede encontrar en parques o jardines urbanos.</string>
<string name="CantCicadatraAtra">Puede cantar desde troncos, ramas o tallos, siempre mirando hacia abajo.</string>
<string name="IdentificacioCicadettaBrevipennis">Cuerpo de unos 18 mm de longitud y coloración negra/parduzca, con los márgenes de los segmentos del abdomen de color naranja. No presenta ningún rasgo fisiológico que permita distinguirla de forma fiable de la Cicadetta cerdaniensis: sólo pueden ser diferenciadas por su canto.</string>
<string name="HabitatCicadettaBrevipennis">Puede habitar ambientes muy diversos, desde llanuras hasta bosques, pero se encuentra básicamente en zonas abiertas de media y alta montaña.</string>
<string name="CantCicadettaBrevipennis">Canta desde tallos o ramas de pocos milímetros de diámetro y desde alturas muy variables.</string>
<string name="IdentificacioCicadettaCerdaniensis">Cuerpo de unos 18 mm de longitud y coloración negra/parduzca, con los márgenes de los segmentos del abdomen de color naranja. No presenta ningún rasgo fisiológico que permita distinguirla de forma fiable de la Cicadetta brevipennis: sólo pueden ser diferenciadas por su canto.</string>
<string name="HabitatCicadettaCerdaniensis">Se puede encontrar en zonas de media y alta montaña, habitualmente en brezales densos pero también en vegetación arbórea. Tienen preferencia por plantas espinosas como el rosal silvestre o la zarza.</string>
<string name="CantCicadettaCerdaniensis">Canta desde tallos o ramas de pocos milímetros de diámetro y desde alturas muy variables.</string>
<string name="IdentificacioHilaphuraVaripes">Coloración negruzca, con cuatro manchas de color crema, dos en el tórax y dos en el abdomen, que forman una cruz negra en medio.</string>
<string name="HabitatHilaphuraVaripes">Habita espacios abiertos, tanto de tierra baja como de media montaña, con clima seco y cálido. Prefiere tierras no cultivadas con poca vegetación o con un claro predominio de la vegetación herbácea por encima de la leñosa.</string>
<string name="CantHilaphuraVaripes">Canta desde los tallos de las plantas herbáceas con pocos milímetros de diámetro.</string>
<string name="IdentificacioLyristesPlebejus">La más grande de las que se puede encontrar en Cataluña, con un cuerpo de unos 35 mm de longitud. Presenta una coloración cenizosa, con un “collar” de color naranja apagado (verdoso cuando sale de la exuvia) y a menudo dos manchas oscuras en las alas anteriores.</string>
<string name="HabitatLyristesPlebejus">Habita principalmente zonas con vegetación elevada y leñosa, pero también se puede encontrar en ambientes no boscosos, como maquias o carrascales poco densos.</string>
<string name="CantLyristesPlebejus">Suele cantar desde ramas gruesas y de forma individual, sin formar coros con otros machos.</string>
<string name="IdentificacioTettigettalnaArgentata">Cuerpo de unos 17 mm de longitud y coloración gris-parduzca, con el vientre claro y dos manchas ocres en la parte superior del abdomen.</string>
<string name="HabitatTettigettalnaArgentata">Puede habitar una gran variedad de comunidades vegetales de tierra baja y media montaña, desde brezales hasta bosques y arboledas. Abunda especialmente en ambientes cálidos y secos con presencia de vegetación leñosa elevada.</string>
<string name="CantTettigettalnaArgentata">Canta sobre todo desde tallos de pocos milímetros de diámetro, pero también puede hacerlo desde ramas o troncos de alturas muy diversas.</string>
<string name="IdentificacioTettigettulaPygmea">La más menuda de las que puede haber en Cataluña, con un cuerpo de unos 13 mm de longitud. Presenta una coloración negra o gris muy oscuro por todo el cuerpo, incluida la zona ventral (a diferencia de la Tettigettalna argentata). También tiene dos manchas de color ocre en la parte superior del abdomen.</string>
<string name="HabitatTettigettulaPygmea">Habita bosques poco densos, que permitan la penetración de la luz del sol, y zonas con vegetación arbustiva elevada, tanto de tierras bajas como de media montaña. Prefiere los ambientes secos y cálidos.</string>
<string name="CantTettigettulaPygmea">Canta desde tallos, ramas o troncos con diámetros que van desde unos pocos milímetros a decenas de centímetros, y lo hacen principalmente desde alturas superiores a los dos metros.</string>
<string name="IdentificacioTibicinaCorsicaFairmairei">Cuerpo de unos 24 mm de longitud, de colores gris oscuro y naranja y con los primeros segmentos del abdomen completamente grises. Igual que la Tibicina tomentosa, tiene los nervios alares de color blanco amarillento en la base y negros en la punta.</string>
<string name="HabitatTibicinaCorsicaFairmairei">Habita ambientes esteparios de zonas bajas, así como carrascales poco densos.</string>
<string name="CantTibicinaCorsicaFairmairei">Canta desde tallos o ramas con un diámetro de entre medio y un centímetro, a menos de dos metros de altitud y desde sitios soleados.</string>
<string name="IdentificacioTibicinaGarricola">Cuerpo de unos 26 mm de longitud, de coloración negra y naranja. Tiene cuatro puntos naranjas en la parte superior del tórax, como la Tibicina quadrisignata, y nervios alares de dos coloraciones: blanco amarillento o verdoso y negro.</string>
<string name="HabitatTibicinaGarricola">Habita ambientes secos y arbustivos de zonas bajas, con especial predilección por la carrasca y la estepa, pero también se puede encontrar en zonas con vegetación arbórea abierta.</string>
<string name="CantTibicinaGarricola">Canta desde el interior de los arbustos, sin que le toque el sol y en apoyos de entre medio y unos pocos centímetros de diámetro.</string>
<string name="IdentificacioTibicinaHaematodes">Cuerpo de unos 30 mm de longitud, con una coloración dominante negra y naranja. Los nervios alares son completamente naranjas, excepto en la forma viridinervis, poco habitual, en que éstos son de color verde y la coloración del cuerpo es ligeramente más apagada.</string>
<string name="HabitatTibicinaHaematodes">Habita zonas de vegetación arbórea o vegetación arbustiva elevada, sobre todo en tierras bajas con clima seco y cálido. Abunda especialmente en encinares, robledales y plantaciones de viña, y también se puede encontrar en parques y jardines urbanos.</string>
<string name="CantTibicinaHaematodes">Canta desde ramas con varios centímetros de diámetro y a una altitud elevada, a menudo cerca de la copa de los árboles.</string>
<string name="IdentificacioTibicinaQuadrisignata">Cuerpo de unos 27 mm de longitud y de color negro o gris muy oscuro, con algunas partes naranjas. Los nervios alares son oscuros y tiene, igual que la Tibicina garricola, cuatro puntos naranjas en el tórax.</string>
<string name="HabitatTibicinaQuadrisignata">Habita bosques abiertos y zonas con vegetación arbustiva, tanto de tierras bajas como de media montaña. Se puede encontrar en encinares, pinedas, alcornocales, olivares o castañares, entre otros, pero también en maquias o landas.</string>
<string name="CantTibicinaQuadrisignata">Canta desde ramas con varios centímetros de diámetro y a una altitud superior a los dos metros, a menudo cerca de la copa de los árboles.</string>
<string name="IdentificacioTibicinaTomentosa">Cuerpo de unos 26 mm de longitud, de color negro o gris muy oscuro y con los bordes de los segmentos del abdomen y las patas de color naranja. Los nervios alares son blancos o de una tonalidad amarillenta en la base y negros en la punta de las alas.</string>
<string name="HabitatTibicinaTomentosa">Habita ambientes abiertos especialmente secos y cálidos, con poca o nula presencia de vegetación arbórea.</string>
<string name="CantTibicinaTomentosa">Canta desde los tallos de las plantas herbáceas con unos pocos milímetros de diámetro.</string>
</resources>

strings.xml (fr):

<resources>
<string name="app_name">CicadaApp</string>

<string name="LesEspecies">Les espèces</string>
<string name="IniciaUnaGravacio">Commencer un enregistrement</string>
<string name="AnalitzaUnFitxer">Analiser un fichier</string>
<string name="title_activity_llista_especies">LlistaEspecies</string>
<string name="title_activity_enregistrador_audio">Enregistrement en cours</string>
<string name="title_activity_resultats">Réesultats</string>
<string name="title_activity_fitxa_especie">Affiche de l\'espèce</string>
<string name="LlistaEspecies">Liste des espèces de cigale avec présence réelle ou potentielle en Catalogne</string>
<string name="Catala">Català</string>
<string name="Castella">Castellano</string>
<string name="Angles">English</string>
<string name="Frances">French</string>
<string name="QuantA">Quant à CicadaApp</string>
<string name="Instruccions">Instructions d\'usage</string>
<string name="Reproduceix">Écouter</string>
<string name="Analitza">Analiser</string>
<string name="Descarta">Écarter</string>
<string name="Atura">Arrêter</string>
<string name="Resultats">Résultats</string>
<string name="ReproduccioEnCurs">Lecture en cours</string>
<string name="SemblancesTrobades">Similarités trouvées</string>
<string name="NoHiHaResultats">Aucune coincidence a été trouvé</string>
<string name="CicadaOrni">Cicada orni</string>
<string name="CicadaBarbaraLusitanica">Cicada barbara lusitanica</string>
<string name="CicadatraAtra">Cicadatra atra</string>
<string name="CicadettaBrevipennis">Cicadetta brevipennis</string>
<string name="CicadettaCerdaniensis">Cicadetta cerdaniensis</string>
<string name="HilaphuraVaripes">Hilaphura varipes</string>
<string name="LyristesPlebejus">Lyristes plebejus</string>
<string name="TettigettalnaArgentata">Tettigettalna argentata</string>
<string name="TettigettulaPygmea">Tettigettula pygmea</string>
<string name="TibicinaCorsicaFairmairei">Tibicina corsica fairmairei</string>
<string name="TibicinaGarricola">Tibicina garricola</string>
<string name="TibicinaHaematodes">Tibicina haematodes</string>
<string name="TibicinaQuadrisignata">Tibicina quadrisignata</string>
<string name="TibicinaTomentosa">Tibicina tomentosa</string>
<string name="CicadaBarbara">Cicada barbara</string>
<string name="TibicinaCorsica">Tibicina corsica</string>
<string name="SspLusitanica">ssp. lusitanica</string>
<string name="SspFairmairei">ssp. fairmairei</string>
<string name="FrequenciaMostreigExigua">* À cause de la baisse fréquence d\'échantillonnage utilisée à l\'enregistrement ou de la présence d\'un filtre de fréquences appliqué par le propre enregistreur, il n\'a pas été possible de vérifier si le spectre fréquentiel du chant coïncide avec celui de cette espèce. Le niveau de similarité montré ne prend pas en compte ce facteur.</string>
<string name="AdvertenciaCicadaBarbaraLusitanica">** Parfois, le chevauchement des chants de plusieurs individus de Cicada orni peut donner l\'impression que le chant enregistré est continu, comme celui de Cicada barbara lusitanica.</string>
<string name="CicadaOrniDeFons">Il s\'a pu détecter ce qui est probablement une ou plusieures Cicada orni en train de chanter de fond.</string>
<string name="Analitzant">Analysant le chant</string>
<string name="title_activity_explorador_fitxers">Seleccionez un fichier</string>
<string name="DirectoriActual">Répertoir courant</string>
<string name="Fitxer">fichier</string>
<string name="Fitxers">fichiers</string>
<string name="FitxerNoValid">Fichier invalide</string>
<string name="SHaDeSeleccionarUnFitxerWav">il faut sélectionner un fichier .wav</string>
<string name="Versio1.0">1.0</string>
<string name="Copyright">Copyright © 2016 \t David Funosas Planas</string>
<string name="DescripcioAplicacio">CicadaApp est une application pour l\'identification automatique de cigales à partir d\'un enregistrement de leur chant. Les espèces incluses sont toutes celles avec présence réelle ou potentielle en Catalogne l\'année 2016.</string>
<string name="DescripcioAplicacio2">La principale utilité de l\'application est l\'identification des cigales du genre Tibicina, pratiquement indiscernables auditivement et pour lesquels les résultats des analyses sont hautement fiables.</string>
<string name="LlicenciaPart1">Cette application est logiciel libre; vous pouvez la redistribuer et/ou modifier sous les termes de la Licence Publique Générale GNU établie par la Free Software Foundation; ou bien la version 2 de la licence ou bien n\'importe quelle version ultérieure.</string>
<string name="LlicenciaPart2">CicadaApp est distribuée avec l\'espoir que ce soit utile, mais sans aucune garantie, pas même la garantie implicite de commercialisation ou adéquation à un but particulier. Lisez la GNU General Public License pour plus de détails.</string>
<string name="DAcord">D\'accord</string>
<string name="Agraiments">Reconnaissances: À Thomas Hertach, Tomi Trilar et Pere Pons pour la provision d\'enregistrements de chants de cigale, et à Gerard Funosas pour le dessin du logo de l\'application, dans tous les cas de forme altruiste et désintéressée</string>
<string name="CorreuContacte">Adresse de contact pour n\'importe quel doute ou consultation sur l\'application: davidfunosas@gmail.com</string>
<string name="MemoriaInsuficient">La mémoire cache actuellement disponible au dispositif est insuffisante pour analyser le chant enregistré. Pour éviter ce problème, c\'est recommandé que les enregistrements soient plus courts.</string>
<string name="ErrorAnalisi">Error en analysant l\'enregistrement</string>
<string name="SenseCoincidencies">Aucune coincidence n\'a été trouvée.</string>
<string name="CantFluix">Le chant enregistré est trop faible pour pouvoir être analysé. Il faudrait enregistrer l\'individu d\'une position plus proche.</string>
<string name="InstruccionsDUs">Instruccions d\'usage</string>
<string name="InstruccionsIConsells">Instruccions et conseils desquels il faut tenir compte au moment d\'utiliser l\'aplication:</string>
<string name="Instruccio1">Pour enregistrer un chant de cigale il faut seulement cliquer à \"Commencer un enregistrement\" et ultérieurement à l\'icône du microphone pour commencer à enregistrer. Le téléphone enregistrera le chant jusqu\'à ce que le microphone soit cliqué de nouveau. Les enregistrements seront emmagasinés au répertoir cicadaAppRecordings avec un nom qui indique le jour et l\'heure de l\'enregistrement.</string>
<string name="Instruccio2">Au moment de faire les enregistrements, c\'est recommandable de signaler l\'individu chanteur avec la part du téléphone où le son est capté et le faire de courte distance (pas plus de 5 mètres), car les enregistreurs des portables sont pensés pour enregistrer dès un écart de juste quelques centimètres de la source du son et souvent ont quelques difficultés au moment d\'enregistrer sons éloignés. Avant l\'analyse, les enregistrements sont amplifiés autant que possible sans dénaturer le son; cependant, une distance supérieure par rapport à l\'individu signifie une pire qualité de l\'enregistrement et, conséquemment, une moindre fiabilité des résultats.</string>
<string name="Instruccio3">Dans le cas qu\'il y aie plus d\'un individu chantant à la fois au moment de l\'enregistrement, soient ils d\'espèces différentes ou de la même, il faut qu\'il n\'y aie un la puissance du chant duquel ressorte clairement sur les autres. Autrement, la superposition des plusieurs chants pourrait dénaturer le résultat. L\'impact causé par la présence d\'autres chants est particulièrement élevé dans le cas de Tettigettalna argentata et Cicadetta cerdaniensis, étant donnée la typologie caractéristique de son chant.</string>
<string name="Instruccio4">Pour éviter que les temps d\'analyse soient élevés, il est conseillé que la duration des enregistrements soit modérée (d\'environ 5-15 secondes).</string>
<string name="Instruccio5">Pour raisons de prudence dans l\'identification, il est récommandé d\'utiliser les niveaux de similarité obtenus par l\'application comme donné d\'appui ou critère additionnel, et ne pas y confier aveuglément, en particulier si en écoutant l\'enregistrement avant de l\'analyser on n\'entend pas le chant nettement.</string>
<string name="LlistaEspeciesCurt">Liste d\'espèces</string>
<string name="Fenologia">Quand on peut la voir</string>
<string name="Identificacio">Identification</string>
<string name="Habitat">Habitat</string>
<string name="Cant">Chant</string>
<string name="PerePons">Photographie de Pere Pons</string>
<string name="PacoFaluke">Photographie de Paco Faluke</string>
<string name="DanielRojas">Photographie de Daniel Rojas</string>
<string name="XavierDeYzaguirre">Photographie de Xavier De Yzaguirre</string>
<string name="MatijaGogala">Photographie de Matija Gogala</string>
<string name="ThomasHertach">Photographie de Thomas Hertach</string>
<string name="AntonioGarciaMaldonado">Photographie d\'Antonio García Maldonado</string>
<string name="CosminOvidiu">Photographie de Cosmin Ovidiu</string>
<string name="JeromeSueur">Photographie de Jérome Sueur</string>
<string name="StephanePuissant">Photographie de Stéphane Puissant</string>
<string name="VincentDerreumaux">Photographie de Vincent Derreumaux</string>
<string name="IgnasiJosepTejedor">Photographie d\'Ignasi Josep Tejedor</string>
<string name="JeanPierreLavigne">Photographie de Jean-Pierre Lavigne</string>
<string name="JoseManuelSesma">Photographie de José Manuel Sesma</string>
<string name="FerranTurmo">Photographie de Ferran Turmo</string>
<string name="MichelBoulard">Photographie de Michel Boulard</string>
<string name="FotografiaResultatsDanielRojas">*La photographie montrée à Résultats est de Daniel Rojas.</string>
<string name="FotografiaResultatsCosminOvidiu">*La photographie montrée à Résultats est de Cosmin Ovidiu.</string>
<string name="FotografiaResultatsVincentDerreumaux">*La photographie montrée à Résultats est de Vincent Derreumaux.</string>
<string name="FotografiaResultatsIvanJesusTorresanoGarcia">*La photographie montrée à Résultats est d\'Iván Jesús Torresano García.</string>
<string name="IdentificacioCicadaBarbaraLusitanica">Corps grise-verdâtre ou gris-brunâtre et avec une dizaine de taches foncées à chaque aile, caractéristique propre du genre. Elle n\'a pas aucun trait physiologique qui permette la distinguer de source sûre de la Cicada orni : elles ne peuvent être différenciées que par leur chant.</string>
<string name="HabitatCicadaBarbaraLusitanica">Son habitat principal est les zones avec végétation arborescente ouverte, spécialement les bois d\'olivier ou caroube ou les pinèdes, mais elle peut aussi habiter zones de végétation arbustive.</string>
<string name="CantCicadaBarbaraLusitanica">Elle peut chanter sur un tronc ou une grosse branche. Quand un mâle commence à chanter, d\'autres mâles proches peuvent le suivre et finir par former un chœur de chants simultanés.</string>
<string name="IdentificacioCicadaOrni">Corps d\'environ 28 mm de longueur et coloration grise-verdâtre ou gris-brunâtre, avec une dizaine de taches foncées à chaque aile. Elle n\'a pas aucun trait physiologique qui permette la distinguer de source sûre de la Cicada orni : elles ne peuvent être différenciées que par leur chant.</string>
<string name="HabitatCicadaOrni">C\'est l\'espèce la plus commune de Catalogne. Elle habite une grande variété de bois et forêts, notamment en abondance dans les pinèdes, mais elle est aussi présente en maquis ou garrigues, à condition qu\'elles soient suffisamment ouvertes pour que les rayons du soleil puissent y pénétrer.</string>
<string name="CantCicadaOrni">Elle peut chanter sur un tronc ou une grosse branche. C\'est habituel entendre plusieurs mâles chantant au même temps, formant un chœur.</string>

<string name="IdentificacioCicadatraAtra">Corps d\'environ 28 mm de longueur et coloration noirâtre, avec deux ou trois taches noires aux ailes antérieures.</string>
<string name="HabitatCicadatraAtra">Elle habite une grande variété d\'environnements, de maquis ou garrigues à forêts à feuilles caduques ou ripisylves, avec préférence par des communautés végétales avec une strate de végétation haute. On peut aussi la trouver dans des parcs ou jardins urbains.</string>
<string name="CantCicadatraAtra">Elle peut chanter sur troncs, branches ou tiges, toujours orientée vers le bas.</string>
<string name="IdentificacioCicadettaBrevipennis">Corps d\'environ 28 mm de longueur et coloration noirâtre/brunâtre, avec les bords des segments de l\'abdomen d\'une couleur orange. Elle n\'a pas aucun trait physiologique qui permette la distinguer de source sûre de la Cicadetta cerdaniensis: elles ne peuvent être différenciées que par leur chant.</string>
<string name="HabitatCicadettaBrevipennis">Elle peut habiter environnements très divers, des plaines aux forêts, mais on peut la trouver basiquement dans les zones ouvertes de moyenne et haute montagne.</string>
<string name="CantCicadettaBrevipennis">Elle chante sur tiges ou branches de seulement quelques millimètres de diamètre et d\'hauteurs très variables.</string>
<string name="IdentificacioCicadettaCerdaniensis">Corps d\'environ 28 mm de longueur et coloration noirâtre/brunâtre, avec les bords des segments de l\'abdomen d\'une couleur orange. Elle n\'a pas aucun trait physiologique qui permette la distinguer de source sûre de la Cicadetta brevipennis: elles ne peuvent être différenciées que par leur chant.</string>
<string name="HabitatCicadettaCerdaniensis">On peut la trouver dans des zones de moyenne et haute montagne, habituellement dans des landes denses, mais aussi dans des zones avec végétation arborescente. Elles ont préférence pour plantes épineuses comme les églantiers ou les ronces.</string>
<string name="CantCicadettaCerdaniensis">Elle chante sur tiges ou branches de seulement quelques millimètres de diamètre et d\'hauteurs très variables.</string>
<string name="IdentificacioHilaphuraVaripes">Couleur noirâtre, avec quatre taches de couleur crème, deux sur le thorax et deux sur l\'abdomen, formant une croix noire au milieu.</string>
<string name="HabitatHilaphuraVaripes">Elle habite les espaces ouverts, tant en basses terres qu\'en moyenne montagne, avec un climat sec et chaud. Elle préfère les terres non cultivées avec peu de végétation ou les zones avec une nette prédominance de la végétation herbacée sur la ligneuse.</string>
<string name="CantHilaphuraVaripes">Elle chante sur les tiges des plantes herbacées avec seulement quelques millimètres de diamètre.</string>
<string name="IdentificacioLyristesPlebejus">La plus grande des cigales qu\'on peut trouver en Catalogne, avec un corps d\'environ 35 mm de longueur. Elle a une couleur cendrée, avec un «collier» orange terne (verdâtre quand elle quitte l\'exuvie) et souvent deux taches foncées sur les ailes antérieures.</string>
<string name="HabitatLyristesPlebejus">Elle habite notamment zones avec végétation haute et boisée, mais on peut aussi la trouver dans des environnements non forestiers, comme maquis ou garrigues ouvertes.</string>
<string name="CantLyristesPlebejus">Elle chante habituellement sur des grosses branques et individuellement, sans former des chœurs avec d\'autres mâles.</string>
<string name="IdentificacioTettigettalnaArgentata">Longueur du corps d\'environ 17 mm, avec une couleur gris-brunâtre, le ventre clair et deux taches ocre sur l\'abdomen.</string>
<string name="HabitatTettigettalnaArgentata">Elle peut habiter une grande variété de communautés végétales en basses terres et en moyenne montagne, de landes aux bois et forêts. Elle abonde notamment dans des environnements chauds et secs avec présence de végétation ligneuse haute.</string>
<string name="CantTettigettalnaArgentata">Elle chante principalement sur tiges de quelques millimètres de diamètre, mais elle peut aussi le faire sur des branches ou troncs d\'hauteurs très différentes.</string>
<string name="IdentificacioTettigettulaPygmea">La plus petite de celles qu\'on peut trouver en Catalogne, avec un corps d\'environ 13 mm de longueur. Elle a une coloration dominante noire ou grise très foncée sur l\'ensemble du corps, y compris la zone ventrale (contrairement à Tettigettalna argentata). Elle a aussi deux taches ocre sur l\'abdomen.</string>
<string name="HabitatTettigettulaPygmea">Elle habite forêts ouvertes, qui permettent la pénétration des rayons du soleil, et zones avec végétation arbustive haute, en basses terres et en moyenne montagne. Elle préfère les environnements secs et chauds.</string>
<string name="CantTettigettulaPygmea">Elle chante sur des tiges, branches ou troncs avec diamètres allant de quelques millimètres à plusieurs dizaines de centimètres, et elle le fait principalement d\'altitudes supérieures aux deux mètres.</string>
<string name="IdentificacioTibicinaCorsicaFairmairei">Corps d\'environ 24 mm de longueur, de couleurs gris foncée et orange et avec les premiers segments de l\'abdomen complètement gris. Comme Tibicina tomentosa, elle a les nervures alaires blanc jaunâtre à la base et noir à l\'apex.</string>
<string name="HabitatTibicinaCorsicaFairmairei">Elle habite environnements steppiques et garrigues ouvertes.</string>
<string name="CantTibicinaCorsicaFairmairei">Elle chante sur tiges ou branches avec un diamètre compris entre un demi-centimètre et un centimètre, notamment d\'altitudes inférieures à deux mètres.</string>
<string name="IdentificacioTibicinaGarricola">Corps d\'environ 26 mm de longueur, de couleurs noire et orange. Elle a quatre points orange sur le thorax, comme Tibicina quadrisignata, et nervures alaires de deux couleurs: noir et blanc jaunâtre ou verdâtre.</string>
<string name="HabitatTibicinaGarricola">Elle habite environnements secs et arbustives en basses terres, avec une prédilection spéciale pour le chêne kermès et les cistes, mais on peut aussi la trouver dans des zones avec végétation arborescente ouverte.</string>
<string name="CantTibicinaGarricola">Elle chante à l\'intérieur des arbustes, sans que les rayons du soleil la touchent et sur un support d\'entre un demi-centimètre et plusieurs centimètres.</string>
<string name="IdentificacioTibicinaHaematodes">Corps d\'environ 30 mm de longueur, avec le noire et l\'orange comme couleurs dominantes. Les nervures alaires sont complètement oranges, sauf pour la forme viridinervis, pas habituelle, cas où ces nervures sont vertes et la couleur du corps est légèrement plus terne.</string>
<string name="HabitatTibicinaHaematodes">Elle habite zones de végétation arborescente ou végétation arbustive haute, surtout dans des basses terres avec un climat sec et chaude. Elle abonde spécialement aux chênaies, rouvraies et plantations de vigne, et on peut aussi la trouver dans des parcs et jardins urbains.</string>
<string name="CantTibicinaHaematodes">Elle chante sur des branches avec plusieurs centimètres de diamètre et d\'une altitude élevée, souvent vers la cime des arbres.</string>
<string name="IdentificacioTibicinaQuadrisignata">Corps d\'environ 27 mm de longueur et de couleur très sombre ou grise, avec quelques parts orange. Les nervures alaires sont foncées et elle a, comme Tibicina garricola, quatre points orange sur le thorax.</string>
<string name="HabitatTibicinaQuadrisignata">Elle habite forêts ouvertes et zones avec végétation arbustive, tant en basses terres qu\'en moyenne montagne. On peut la trouver dans des chênaies, pinèdes, subéraies, oliveraies ou châtaigneraies, entre d\'autres, mais aussi dans des maquis ou landes.</string>
<string name="CantTibicinaQuadrisignata">Elle chante sur des branches avec plusieurs centimètres de diamètre et d\'une altitude supérieure aux deux mètres, souvent près de la cime des arbres.</string>
<string name="IdentificacioTibicinaTomentosa">Corps d\'environ 26 mm de longueur, noir ou gris très foncé et avec les pattes et les bords des segments de l\'abdomen de couleur orange. Les nervures alaires sont blanches ou jaunâtres à la base et noires à l\'apex.</string>
<string name="HabitatTibicinaTomentosa">Elle habite environnements ouverts particulièrement secs et chauds, avec faible ou nulle présence de végétation arborescente.</string>
<string name="CantTibicinaTomentosa">Elle chante sur des tiges de plantes herbacées avec seulement quelques millimètres de diamètre.</string>
</resources>