



Exercicis de programació

Autor:
Marc López

Aquest dossier recull un conjunt d'exercicis en Java i C++ per practicar els coneixements adquirits durant el curs

12 d'agost de 2012

1	Estructures dinàmiques	5
1.1	Exercici 1 - Afegir enters a una pila dinàmica	5
1.2	Exercici 2 - Afegir final d'un estructura dinàmica amb sentinella final. . .	6
1.3	Exercici 3 - Afegir final d'una estructura dinàmica final	7
1.4	Exercici 4 - Afegir Inici i Final d'una estructura dinàmica	9
1.5	Exercici 5 - Afegir final d'una estructura dinàmica circular	10
2	Rekursivitat	13
2.1	Exercici 1 - Càlcul del factorial	13
2.2	Exercici 2 - Joc del Pescamines	13
2.3	Exercici 3 - Simulació de l'erupció d'un volcà	14
3	Cerques	17
3.1	Exercici 1 - Cerca d'un número imparell a dins d'una columna	17
3.2	Exercici 2 - Cercar el primer múltiple de N dins de la matriu	17
3.3	Exercici 3 - Cercar el primer múltiple de N dins d'una matriu de diferents mides	18

ACTIVITAT 1

ESTRUCTURES DINÀMIQUES

1.1 Exercici 1 - Afegir enters a una pila dinàmica

Es disposa d'una pila d'enters amb representació dinàmica (estructura simplement encadenada amb un punter a l'inici) tal i com es mostra a la figura que hi ha al final del text. Implementar en C els mètodes de la pila (constructor per defecte, **Buida**, **Empila**, **Desempila** i **Cim**).

Crear un algoritme que tracti una seqüència d'enters acabada en 0 de manera que si es llegeix un valor positiu, aquest valor es posi a la pila i si el valor és negatiu es tregui un element de la pila. En ambdós casos, cal mostrar el valor que hi ha al cim de la pila després de fer l'operació (excepte en el cas que la pila no tingui elements que cal mostrar un asterisc, *).

En acabar la seqüència, cal buidar la pila element a element mostrant els valors que hi ha al cim de la pila després de treure cada element (amb el corresponent asterisc final). Per facilitar la lectura de la sortida cal separar els valors de sortida amb un espai en blanc entre ells i finalitzar la sortida amb un final de línia.

El fitxer de capçalera és el següent (**pilaDinamica.h**):

```
#ifndef TAD_pilaDinamica_h
#define TAD_pilaDinamica_h

struct node {
    int dada;
    node *seguent;
};

class pilaDinamica { // pila dinamica
    node * inici;
public:
    pilaDinamica();
    bool Buida() const;
    void Empila(int i);
    void Desempila();
    int Cim() const;
};
#endif
```

Un esquelet de programa principal seria (**main.cpp**):

```
#include <iostream>
```

```
#include "pilaDinamica.h"
using namespace std;

int main() {
    pilaDinamica p;
    int n;

    cin >> n;
    while (n != 0) {
        // processar valor
        // mostrar valor o *
        cout << ' ';
        cin >> n;
    }

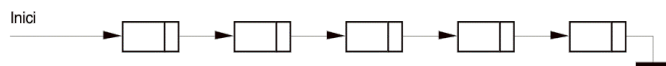
    // buidar la pila
    // mostrant valor o *
    cout << endl;
}
```

Exemples de funcionament:

Entrada: 1 2 3 4 -5 -6 -7 -8 0
Sortida: 1 2 3 4 3 2 1 *

Entrada: 1 2 3 4 -1 0
Sortida: 1 2 3 4 3 2 1 *

Entrada: 1 -1 2 -2 3 -3 0
Sortida: 1 * 2 * 3 *



Disposem dels adjunts següents a la plataforma ACME: pilaDinamica.h

1.2 Exercici 2 - Afegir final d'un estructura dinàmica amb sentinella final.

Es disposa d'una estructura dinàmica simplement encadenada amb sentinella final que disposa d'un punter a l'inici i un altre al final tal i com es representa a la figura que hi ha al final del text. Implementar en C el mètode **AfegirFinal**. Cal tenir en compte que el sentinella s'ha d'usar en totes les cerques (cerca amb sentinella) i no s'ha de modificar el seu valor a la resta d'operacions. Usar l'operació anterior en el següent programa (**main.cpp**):

```
#include <iostream>
#include "estructuraDinamica.h"
using namespace std;

void main() {
    estructuraDinamica e;
    int n;

    cin >> n;
    while (n!=0) {
        e.AfegirFinal(n);
        cin >> n;
    }
}
```

```

    }
    e.Llistar();
}

```

Cal tenir en compte que `AfegirFinal` no verifica mai si ja existeixen els elements. El fitxer de capçalera és el següent (`estructuraDinamica.h`):

```

#ifndef TAD_estructuraDinamicaSentinellaFinal_h
#define TAD_estructuraDinamicaSentinellaFinal_h

struct node {
    int dada;
    node *seguent;
};

class estructuraDinamica { // Sentinella Final
    node * inici, * final;
public:
    estructuraDinamica();
    void AfegirFinal(int i);
    void Llistar() const;
};
#endif

```

El constructor per defecte i el mètode `Llistar` són els següents:

```

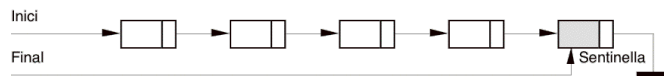
estructuraDinamica::estructuraDinamica() {
    inici = final = new node;
    inici->seguent = NULL; inici->dada = 0;
}

void estructuraDinamica::Llistar() const {
    node * p = inici;

    while (p != final) {
        cout << p->dada << " ";
        p = p->seguent;
    }
    cout << "S:" << p->dada << endl; // llista el sentinella
}

```

Cal descarregar el codi adjunt al problema.



Disposem dels adjunts següents a la plataforma ACME: `main.cpp`, `estructuraDinamica.h`, `estructuraDinamica.cpp`

1.3 Exercici 3 - Afegir final d'una estructura dinàmica final

Es disposa d'una estructura dinàmica simplement encadenada amb un punter a l'inici i un altre al final tal i com es representa a la figura que hi ha al final del text. Implementar en C el mètode `AfegirFinal` i usar-lo amb el següent programa (`main.cpp`):

```

#include <iostream>
#include "estructuraDinamica.h"

```

```
using namespace std;

void main() {
    estructuraDinamica e;
    int n;

    cin >> n;
    while (n != 0) {
        e.AfegirFinal(n);
        cin >> n;
    }
    e.Llistar();
}
```

Cal tenir en compte que `AfegirFinal` no verifica mai si ja existeixen els elements. El fitxer de capçalera és el següent (`estructuraDinamica.h`):

```
#ifndef TAD_estructuraDinamicaFinal_h
#define TAD_estructuraDinamicaFinal_h

struct node {
    int dada;
    node *seguent;
};

class estructuraDinamica { // Final
    node * inici, * final;
public:
    estructuraDinamica();
    estructuraDinamica(estructuraDinamica & e);
    void AfegirFinal(int i);
    void Llistar() const;
};
#endif
```

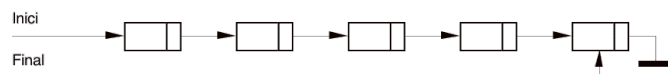
El constructor per defecte i el mètode `Llistar` són els següents:

```
estructuraDinamica::estructuraDinamica() {
    inici = final = NULL;
}

void estructuraDinamica::Llistar() const {
    node * p = inici;

    while (p != final) {
        cout << p->dada << " ";
        p = p->seguent;
    }
    if (p != NULL) cout << p->dada << endl;
    else cout << endl;
}
```

Cal descarregar el codi adjunt al problema.



Disposem dels adjunts següents a la plataforma ACME: `estructuraDinamica.cpp`, `estructuraDinamica.h`, `main.cpp`

1.4 Exercici 4 - Afegir Inici i Final d'una estructura dinàmica

Es disposa d'una estructura dinàmica simplement encadenada amb un punter a l'inici tal i com es representa a la figura que hi ha al final del text. Implementar en C els mètodes **AfegirInici** i **AfegirFinal** per usar-los amb el següent programa (**main.cpp**):

```
#include <iostream>
#include "estructuraDinamica.h"
using namespace std;

void main() {
    estructuraDinamica e;
    int n;

    cin >> n;
    while (n != 0) {
        if (n > 0)
            e.AfegirFinal(n);
        else
            e.AfegirInici(n);
        cin >> n;
    }
    e.Llistar();
}
```

Cal tenir en compte que **AfegirInici** i **AfegirFinal** no verifiquen mai si ja existeixen els elements. El fitxer de capçalera és el següent (**estructuraDinamica.h**):

```
#ifndef TAD_estructuraDinamica_h
#define TAD_estructuraDinamica_h

struct node {
    int dada;
    node *seguent;
};

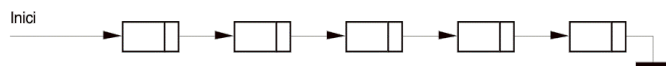
class estructuraDinamica { // Estructura dinamica
    node * inici;
public:
    estructuraDinamica();
    void AfegirInici(int i);
    void AfegirFinal(int i);
    void Llistar() const;
};
#endif
```

El constructor per defecte i el mètode **Llistar** són els següents:

```
estructuraDinamica::estructuraDinamica() {
    inici = NULL;
}

void estructuraDinamica::Llistar() const {
    node * p = inici;

    while (p != NULL) {
        cout << p->dada << " ";
        p = p->seguent;
    }
    cout << endl;
}
```



Cal descarregar el codi adjunt al problema.

Disposem dels adjunts següents a la plataforma ACME: `estructuraDinamica.cpp`, `estructuraDinamica.h`, `main.cpp`

1.5 Exercici 5 - Afegir final d'una estructura dinàmica circular

Es disposa d'una estructura dinàmica circular simplement encadenada amb un punter a l'inici tal i com es representa a la figura que hi ha al final del text. Implementar en C el mètode **AfegirFinal** i usar-lo amb el següent programa (`main.cpp`):

```
#include <iostream>
#include "estructuraDinamica.h"
using namespace std;

void main() {
    estructuraDinamica e;
    int n;

    cin >> n;
    while (n!=0) {
        e.AfegirFinal(n);
        cin >> n;
    }
    e.Llistar();
}
```

Cal tenir en compte que **AfegirFinal** no verifica mai si ja existeixen els elements. El fitxer de capçalera és el següent (`estructuraDinamica.h`):

```
#ifndef TAD_estructuraCircular_h
#define TAD_estructuraCircular_h

struct node {
    int dada;
    node *seguent;
};

class estructuraDinamica { // Circular
    node * inici;
public:
    estructuraDinamica();
    void AfegirFinal(int i);
    void Llistar() const;
};
#endif
```

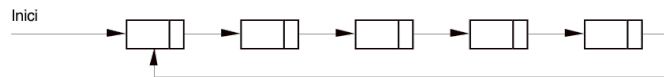
El constructor per defecte i el mètode **Llistar** són els següents:

```
estructuraDinamica::estructuraDinamica() {
    inici = NULL;
}

void estructuraDinamica::Llistar() const {
    if (inici != NULL) {
```

```
node * p = inici->seguent; cout << inici->dada << " ";  
while (p != inici) {  
    cout << p->dada << " "; p = p->seguent;  
}  
cout << endl;  
}
```

Cal descarregar el codi adjunt al problema.



Disposem dels adjunts següents a la plataforma ACME: estructuraDinamica.cpp, main.cpp, estructuraDinamica.h

ACTIVITAT 2

RECURSIVITAT

2.1 Exercici 1 - Càlcul del factorial

Fes un algorisme/programa en Java que donat un número natural ens calculi els seu factorial.

NOTA: La presentació per pantalla ha de ser com es mostra a continuació:

ENTRAR NUMERO:

4

FACTORIAL=24

ENTRAR NUMERO:

0

FACTORIAL=1

ENTRAR NUMERO:

1

FACTORIAL=1

ENTRAR NUMERO:

-1

ERROR

2.2 Exercici 2 - Joc del Pescamines

Farem un programa en Java que permeti jugar al joc del Pescamines.

El terreny inicial serà un terreny rectangular de 1 a 26 files representades per les lletres minúscules (a..z), i de 1 a 26 per les columnes representades per les lletres majúscules (A..Z). Caldrà triar el nombre de mines a inserir en el terreny, com a mínim 1 i com a màxim el nombre del caselles del terreny.

La ubicació de les mines es realitzarà de manera aleàtoria, primer calculant la fila i després la columna. Vigileu de no posar una mina en una casella ja ocupada.

El joc consisteix en destapar totes les caselles que no amaguin una mina, i en posar una bandera a les caselles en què penseu que hi ha una mina.

Si és descobreix una casella amb un número, aquest indica les mines que sumen totes les caselles circumdants.

Si es descobreix una casella sense número ens indica que cap de les caselles veïnes té mina i llavors aquestes es descobreixen automàticament.

Si es descobreix una casella amb mina es perd la partida.

Es guanya si s'aconsegueix descobrir totes las casellas sense mines, i les caselles amb mines degudament marcades amb una bandera.

JOC DEL PESCAMINES
ENTRA LLAVOR: 3
NUMERO DE FILES (DE 1 A 26): 2

|JOC DEL PESCAMINES
|ENTRA LLAVOR: 2
|NUMERO DE FILES (DE 1 A 26): 1

```

NUMERO DE COLUMNES (DE 1 A 26): 5
NUMERO DE MINES (DE 1 A 10): 3
PESCAMINES (BOMBES:3 BANDERES:0 CASELLES:10 DESTAPADES:0)
  ABCDE
a -----
b -----
ENTRA FILA,COLUMNA,OPCIO(1 DESTAPAR,2 BANDERA, 3 TREURE BANDERA):aA1
PESCAMINES (BOMBES:3 BANDERES:0 CASELLES:10 DESTAPADES:1)
  ABCDE
a 2-----
b -----
ENTRA FILA,COLUMNA,OPCIO(1 DESTAPAR 2 BANDERA, 3 TREURE BANDERA):aB1
PESCAMINES (BOMBES:3 BANDERES:0 CASELLES:10 DESTAPADES:1)
  ABCDE
a 2x---
b -----
HO SENTO, T'HA PETAT UNA MINA!

-----
JOC DEL PESCAMINES
ENTRA LLAVOR: 10
NUMERO DE FILES (DE 1 A 26): 3
NUMERO DE COLUMNES (DE 1 A 26): 8
NUMERO DE MINES (DE 1 A 24): 2
PESCAMINES (BOMBES:2 BANDERES:0 CASELLES:24 DESTAPADES:0)
  ABCDEFGH
a -----
b -----
c -----
ENTRA FILA,COLUMNA,OPCIO (1 DESTAPAR, 2 BANDERA, 3 TREURE BANDERA): aC1
PESCAMINES (BOMBES:2 BANDERES:0 CASELLES:24 DESTAPADES:1)
  ABCDEFGH
a --1-----
b -----
c -----
ENTRA FILA,COLUMNA,OPCIO (1 DESTAPAR, 2 BANDERA, 3 TREURE BANDERA): aB1
PESCAMINES (BOMBES:2 BANDERES:0 CASELLES:24 DESTAPADES:13)
  ABCDEFGH
a 1-----
b 112---
c 1---
ENTRA FILA,COLUMNA,OPCIO (1 DESTAPAR, 2 BANDERA, 3 TREURE BANDERA): aD2
PESCAMINES (BOMBES:2 BANDERES:1 CASELLES:24 DESTAPADES:13)
  ABCDEFGH
a 1?----
b 112---
c 1---
ENTRA FILA,COLUMNA,OPCIO (1 DESTAPAR, 2 BANDERA, 3 TREURE BANDERA): aH1
PESCAMINES (BOMBES:2 BANDERES:1 CASELLES:24 DESTAPADES:19)
  ABCDEFGH
a 1?--1
b 112-1
c 1-1
ENTRA FILA,COLUMNA,OPCIO (1 DESTAPAR, 2 BANDERA, 3 TREURE BANDERA): bF2
PESCAMINES (BOMBES:2 BANDERES:2 CASELLES:24 DESTAPADES:19)
  ABCDEFGH
a 1?--1
b 112?1
c 1-1
ENTRA FILA,COLUMNA,OPCIO (1 DESTAPAR, 2 BANDERA, 3 TREURE BANDERA): aE1
PESCAMINES (BOMBES:2 BANDERES:2 CASELLES:24 DESTAPADES:20)
  ABCDEFGH
a 1?2-1
b 112?1
c 1-1
ENTRA FILA,COLUMNA,OPCIO (1 DESTAPAR, 2 BANDERA, 3 TREURE BANDERA): aF1
PESCAMINES (BOMBES:2 BANDERES:2 CASELLES:24 DESTAPADES:21)
  ABCDEFGH
a 1?211
b 112?1
c 1-1
ENTRA FILA,COLUMNA,OPCIO (1 DESTAPAR, 2 BANDERA, 3 TREURE BANDERA): cF1
PESCAMINES (BOMBES:2 BANDERES:2 CASELLES:24 DESTAPADES:22)
  ABCDEFGH
a 1?211
b 112?1
c 111
ENHORABONA, HAS NETEJAT EL TERRENY!

```

2.3 Exercici 3 - Simulació de l'erupció d'un volcà

Volem simular amb Java l'erupció d'un volcà, i concretament l'expansió de la lava a partir del cràter. La lava s'ha de desplaçar, des d'una Posició, a les posicions adjacents que no estiguin ja cobertes i que siguin més baixes o iguals que la Posició original. Per modelitzar-ho disposem d'una classe *ZonaVolcànica*, que no serà res més que una matriu de posicions. Vegem-ne els seus atributs:

```

a_t: taula[1..MAXF][1..MAXC] de ^Posicio
a_nf: enter // nombre de files
a_nc: enter // nombre de columnes

```

Pel que fa a la classe Posició, tindrà mètodes que permetran obtenir l'altura sobre el nivell del mar i també si està ja coberta de lava o no. També hi haurà un mètode modificador que cobrirà de lava la posició. Vegem-ho:

```

consultors
    metode coberta() retorna b:boolean
    metode altura() retorna a:enter
fconsultors

modificadors
    metode cobrir()
fmodificadors

```

Heu d'implementar el mètode modificador de la classe ZonaVolcànica. Heu d'utilitzar recursivitat.

```

metode erupcio(f: enter, c:enter)
{P:: (f,c dins del rang) i (a_t[f,c] no coberta de lava)}
{Q:: (a_t[f,c] coberta) i (totes les posicions a les que pot arribar la lava
    des de a_t[f,c] tambe cobertes)}

```

Per exemple, si volem simular una erupció amb el cràter a la posició (5,7), la crida haurà de ser simplement: erupcio(5,7). Feu una classe AplVolca que contingui el mètode main que ens permeti testear el funcionament de la classe ZonaVolcànica i el seu mètode erupció.

NOTA: La presentació per pantalla ha de ser com es mostra a continuació

```

NUMERO FILES:3
NUMERO COLUMNES:3
ALTURA POSICIO(0,0):30
ALTURA POSICIO(0,1):29
ALTURA POSICIO(0,2):23
ALTURA POSICIO(1,0):26
ALTURA POSICIO(1,1):25
ALTURA POSICIO(1,2):22
ALTURA POSICIO(2,0):27
ALTURA POSICIO(2,1):28
ALTURA POSICIO(2,2):21
NUMERO FILA ERUPCIO:1
NUMERO COLUMNA ERUPCIO:1
VOLCA:
  0 1 2
0   *
1  * *
2   *

```


ACTIVITAT 3

CERQUES

3.1 Exercici 1 - Cerca d'un número imparell a dins d'una columna

Fes un algorisme/programa en Java que permeti entrar una matriu quadrada 3 x 3. A continuació ens ha de demanar cada un dels valors d'aquesta matriu i ens ha de demanar un número de columna. Volem que ens digui si en la columna seleccionada com a mínim hi ha algun número imparell i quin és.

NOTA: La presentació per pantalla ha de ser com es mostra a continuació

```
ENTRAR ELEMENT FILA 1 COLUMNA 1
10
ENTRAR ELEMENT FILA 1 COLUMNA 2
15
ENTRAR ELEMENT FILA 1 COLUMNA 3
21
ENTRAR ELEMENT FILA 2 COLUMNA 1
15
ENTRAR ELEMENT FILA 2 COLUMNA 2
100
ENTRAR ELEMENT FILA 2 COLUMNA 3
25
ENTRAR ELEMENT FILA 3 COLUMNA 1
20
ENTRAR ELEMENT FILA 3 COLUMNA 2
25
ENTRAR ELEMENT FILA 3 COLUMNA 3
1000
QUINA COLUMNA? 3
HI HA UN NUMERO IMPARELL QUE ES EL 21.
```

En el cas de que no hi hagi cap numero parell haura d'escriure el missatge

EN LA COLUMNA SELECCIONADA NO HI HA CAP NUMERO IMPARELL.

3.2 Exercici 2 - Cercar el primer múltiple de N dins de la matriu

Fes un algorisme/programa en Java que donada una matriu quadrada de 3*3 de números enters ens ha de dir quina és la posició del primer múltiple del número escollit per l'usuari.

NOTA: La presentació per pantalla ha de ser com es mostra a continuació

```
ENTRAR ELEMENT FILA 1 COLUMNA 1
10
ENTRAR ELEMENT FILA 1 COLUMNA 2
15
ENTRAR ELEMENT FILA 1 COLUMNA 3
21
ENTRAR ELEMENT FILA 2 COLUMNA 1
15
ENTRAR ELEMENT FILA 2 COLUMNA 2
100
ENTRAR ELEMENT FILA 2 COLUMNA 3
25
ENTRAR ELEMENT FILA 3 COLUMNA 1
20
ENTRAR ELEMENT FILA 3 COLUMNA 2
25
ENTRAR ELEMENT FILA 3 COLUMNA 3
1000
```

```
NUMERO?  
21  
PRIMER MULTIPLE TROBAT A LA FILA 1 COLUMN 3
```

En cas de que no hi hagi cap múltiple el missatge ha de ser:

```
NO HI HA CAP MULTIPLE
```

3.3 Exercici 3 - Cercar el primer múltiple de N dins d'una matriu de diferents mides

Fes un algorisme/programa en Java que donada una matriu de N files i M columnes de números enters, ens ha de dir quina és la posició del primer múltiple del número escollit per l'usuari. Les matrius tindran com a màxim 10 files i 10 columnes.

NOTA: La presentació per pantalla ha de ser com es mostra a continuació

```
NUMERO DE FILES?  
3  
NUMERO DE COLUMNES?  
3  
ENTRAR ELEMENT FILA 1 COLUMN 1  
10  
ENTRAR ELEMENT FILA 1 COLUMN 2  
15  
ENTRAR ELEMENT FILA 1 COLUMN 3  
21  
ENTRAR ELEMENT FILA 2 COLUMN 1  
15  
ENTRAR ELEMENT FILA 2 COLUMN 2  
100  
ENTRAR ELEMENT FILA 2 COLUMN 3  
25  
ENTRAR ELEMENT FILA 3 COLUMN 1  
20  
ENTRAR ELEMENT FILA 3 COLUMN 2  
25  
ENTRAR ELEMENT FILA 3 COLUMN 3  
1000  
NUMERO?  
21  
PRIMER MULTIPLE TROBAT A LA FILA 1 COLUMN 3
```

En cas de que no hi hagi cap múltiple el missatge ha de ser:

```
NO HI HA CAP MULTIPLE
```