METODOLOGIA I TECNOLOGIA DE LA **PROGRAMACIÓ 2** (GEINF, GDDV i GEB)

Anton Bardera, Miquel Feixas, Jaume Rigau,
Joan Surrell, Màrius Vila, Pau Xiberta

Curs 2019-20

Àrea LSI, Departament IMAE, Universitat de Girona

Sessions de laboratori

S1. Introducció i conceptes previs

S2. Introducció als objectes

S3. Objectes compostos

S4. Taules d'objectes

S5. Pràctica d'objectes completa

S6. Punters

S7. Cua

S8. Llista ordenada

S9. Taules dinàmiques

S10. Matrius dinàmiques – Pràctica final

S11. Recursivitat

S12. Pràctica final

METODOLOGIA I TECNOLOGIA DE LA **PROGRAMACIÓ 2**

Sessió 11: Recursivitat

Anton Bardera, Miquel Feixas, Jaume Rigau,
Joan Surrell, Màrius Vila, Pau Xiberta

Curs 2019-20

Àrea LSI, Departament IMAE, Universitat de Girona

Continguts

- **Recursivitat**
- Activitats de l'ACME

Continguts

- **Recursivitat**
  - Concepte
  - Exemples
  - Recursivitat a dins de classes
- Activitats ACME

Recursivitat

- Una acció/funció recursiva és aquella que es crida a ella mateixa:
  - es genera una seqüència o arbre de crides
  - cal arribar sempre al cas (o casos) base (no genera crida)
- Recursivitat i iteració:
  - el gran avantatge de la recursivitat és la creació d'algorismes normalment més simples que les seves versions iteratives
  - el seu desavantatge és l'increment de recursos respecte a les seves versions iteratives
- Recursivitat i classes:
  - els mètodes de les classes poden ser recursius
  - …però sol ser necessari disposar d'un segon mètode privat per adaptar paràmetres

Exemple recursivitat

```
#include <iostream>
using namespace std;

// Pre: n és el nombre de discs (n≥0); from, to i aux són els noms de les piles
// Post: solució de les torres d'Hanoi movent n discs de from a to usant aux
void Hanoi(int n, char from, char to, char aux) {
  if (n > 0) {
    Hanoi(n-1, from, aux, to);
    cout << "Move disk from " << from << " to " << to << endl;
    Hanoi(n-1, aux, to, from);
  }
}

int main() {
  Hanoi(3, 'a', 'c', 'b');
  return 0;
}
```

Exemple recursivitat

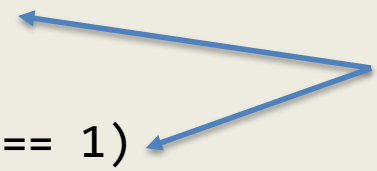
- Càlcul de x^y amb x un enter i y un enter més gran o igual a 0

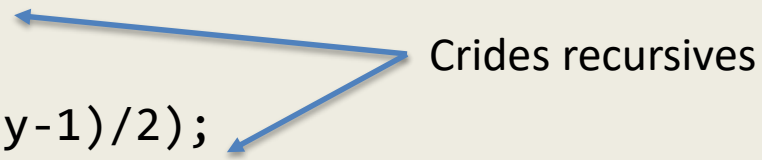
- **Solució trivial**:
  - Usant la propietat $x^y = x \cdot x^{y-1}$
- **Solució més eficient** (per valors grans de y):
  - Si y és 0: $x^y = 1$
  - Si y és 1: $x^y = x$
  - Si y és parell: $x^y = (x \cdot x)^{y/2}$
  - Si y és senar: $x^y = x \cdot (x \cdot x)^{(y-1)/2}$

Exemple recursivitat

```
#include <iostream>
using namespace std;

// Pre: y>=0
// Post: retorna el resultat de x^y
int elevar(int x, int y) {
  if (y == 0)
    return 1;
  else if (y == 1)
    return x;
  else if (y % 2 == 0)
    return elevar(x*x, y/2);
  else
    return x * elevar(x*x, (y-1)/2);
}
```





Recursivitat dins classes

- La recursivitat també la podrem usar dins les classes
- Senzillament haurem de cridar els mètodes recursius dins els propis mètodes de la classe
- Típicament els mètodes immersius els declararem com a privats
- Els atributs no els passarem mai com a paràmetres, ja que també els tenim disponibles en els mètodes recursius i així necessitem menys recursos

Exemple recursivitat

```
// Pre: Les dates estan ordenades creixentment
// Post: retorna -1 quan d no hi és;
//  retorna el valor de la posició >= 0 de d quan d hi és
int TaulaData::cercaDicotomica(Data d) const {
  return icercaDicot(d, 0, a_n-1);
}

// Pre: 0 <= ini <= fi+1 <= nombre d'elements i
//      d no hi és a 0..ini-1 ni de fi+1 al final
// Post: retorna -1 quan d no hi és;
//  retorna el valor de la posició >= 0 de d quan d hi és
int TaulaData::iCercaDicot(Data d, int ini, int fi) const {
  int pos, mig;
  if (ini > fi) {
    pos = -1;
  }
  else { // ini <= fi
    mig = (ini + fi) / 2;
    if (a_t[mig] == d)
      pos = mig;
    else if (a_t[mig] > d)
      pos = iCercaDicot(d, ini, mig-1);
    else // a_t[mig] < d
      pos = iCercaDicot(d, mig+1, fi);
  }
  return pos;
}
```

TaulaData.cpp

```
class TaulaData {
  public:
    TaulaData();

    int cercaDicotomica(Data) const;
  private:
    static const int MAX = 100;
    int a_n;
    Data a_t[MAX];

    int iCercaDicot(Data, int, int) const;
};
```

TaulaData.h

```
#include <iostream>
#include "TaulaData.h"
int main() {
  Data d1;
  TaulaData td;

  int pos = td.cercaDicotomica(d1);
  cout << "La data es troba a la posicio: "
       << pos << endl;
}
```

main.cpp

Exemple recursivitat

TaulaEnter.cpp

```
void TaulaEnter::mergesort() {
  i_mergesort(0, a_n-1);
}

void TaulaEnter::i_mergesort(int esq, int dre) {
  if (esq < dre) {
    int mig = (esq + dre) / 2;
    i_mergesort(esq, mig);
    i_mergesort(mig+1, dre);
    fusio(esq, mig+1, dre);
  }
}

void TaulaEnter::fusio(int esq, int ini2, int dre) {
  ...
}
```

TaulaEnter.h

```
class TaulaEnter {
  public:
    static const int MAX = 100;
    TaulaEnter();
    void mergesort();
  private:
    int a_n;
    int a_t[MAX];

    void i_mergesort(int esq, int dre);
    void fusio(int esq, int ini2, int dre);
};
```

main.cpp

```
#include <iostream>
#include "TaulaEnter.h"
int main() {
  TaulaEnter t;
  ...
  t.mergesort();
}
```

Exemple recursivitat

TaulaEnter.cpp

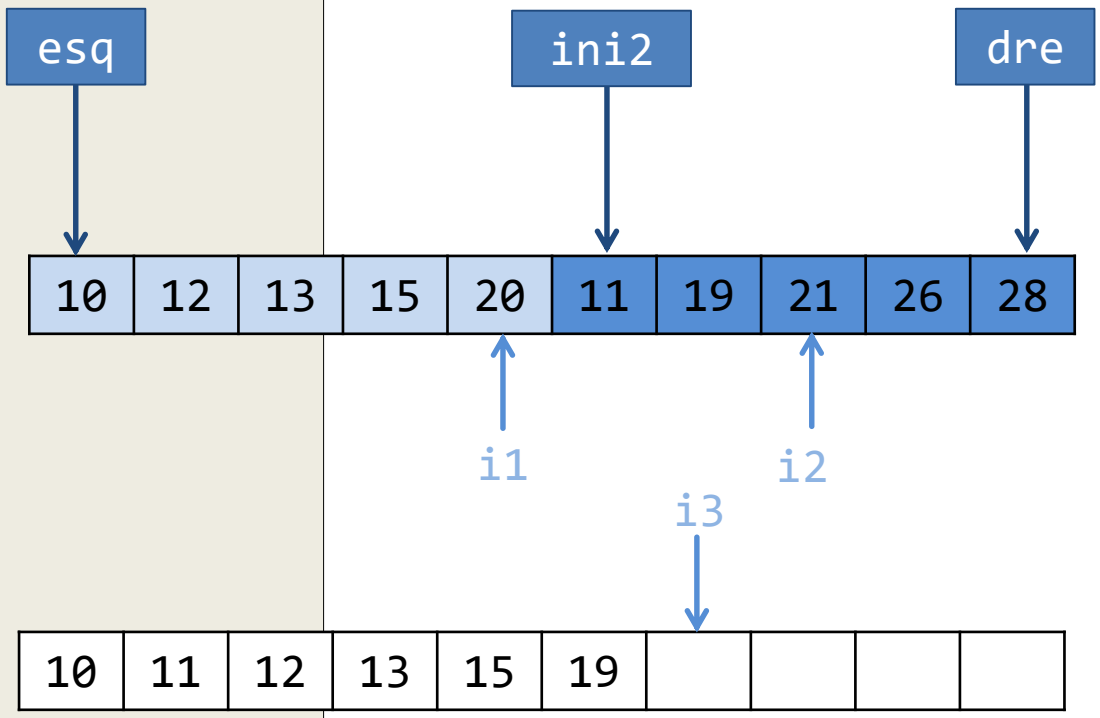
```
void TaulaEnter::fusio(int esq, int ini2, int dre) {
  int aux[MAX]; // taula auxiliar
  int i1=esq, i2=ini2, i3=esq;

  while (i1<=ini2-1 and i2<=dre) {
    if (a_t[i1] < a_t[i2]) {
      aux[i3] = a_t[i1]; i3++; i1++;
    }
    else {
      aux[i3] = a_t[i2]; i3++; i2++;
    }
  }

  while (i1<=ini2-1) {
    aux[i3] = a_t[i1]; i3++; i1++;
  }

  while (i2<=dre) {
    aux[i3] = a_t[i2]; i3++; i2++;
  }

  for (int i = esq; i<=dre; i++)
    a_t[i] = aux[i];
}
```



Continguts

- Recursivitat
- **Activitats ACME**

ACME sessió 11

- En aquesta sessió hi ha 4 exercicis.
- Els 3 primers valen cadascun un 30% de la nota mentre que el darrer val el 10% restant
- No cal crear classes ni objectes ni treballar amb memòria dinàmica
- Cal llegir les dades, invocar l'algorisme recursiu i mostrar el resultat

Data de lliurament: 10 dies a partir de la sessió

ACME, exercici 1 (és potència de 2?)

- Implementar una funció recursiva que determini si un valor enter positiu és o no potència de 2.
- Per poder treballar amb valors de major rang, cal utilitzar el tipus `long long` i no els `int` que es fan servir habitualment.

- Exemple d'execució:

```
ENTRA UN VALOR ENTER POSITIU (0 PER ACABAR):
-1
ENTRA UN VALOR ENTER POSITIU (0 PER ACABAR):
3
3 NO ES POTENCIA DE 2
ENTRA UN VALOR ENTER POSITIU (0 PER ACABAR):
1024
1024 ES POTENCIA DE 2
ENTRA UN VALOR ENTER POSITIU (0 PER ACABAR):
0
```

ACME, exercici 2 (separador de milers)

- Dissenyar una acció recursiva en C++ tal que, donat un valor enter positiu escrigui aquest valor amb separadors de milers (caràcter '.').
  - Per poder treballar amb valors de major rang, cal utilitzar el tipus `long long` i no els `int` que es fan servir habitualment.
  - Per omplir una sortida amb 0 cal posar `setw(3)` i `setfill('0')` en el `cout`.

- Exemple d'execució del programa:

```
ENTRA UN ENTER POSITIU:
999999
QUANTS VALORS VOLS MOSTRAR (N>0)?
4
VALORS AMB SEPARADORS DE MILERS:
999.999 1.000.000 1.000.001 1.000.002
```

ACME, exercici 3 (suma de caselles anteriors)

- Dissenyeu una funció recursiva que indiqui si en una taula d'enters hi ha una casella que és la suma de totes les anteriors.
  - Cal retornar l'índex de la casella suma
  - En cas que hi hagi més d'una casella, s'ha de mostrar la primera.
  - Pot donar-se el cas que no n'hi hagi cap.
  - Suggeriment: cal fer una immersió
- La presentació per pantalla ha de ser com es mostra a continuació:

```
ENTRA EL NOMBRE DE CASELLES (>0):
7
ENTRA ELS VALORS:
1 23 14 -7 31 -2 9
S'HA TROBAT UNA CASELLA:
1 + 23 + 14 - 7 = 31
```

- Si no es troba cap casella que compleixi aquesta condició, es mostra `NO S'HA TROBAT CAP CASELLA`

ACME, exercici 4 (triangle de Sierpinski)

- Realitzar un programa que mostri per pantalla una figura fractal.
  - Es demanarà per pantalla el nivell de la figura i el caràcter que s'usarà per dibuixar.
  - Un cop entrats els dos valors el programa mostrarà el fractal per pantalla.
  - Mirant els tres exemples adjunts es pot veure que un fractal de nivell n es defineix com 3 fractals de nivell n-1.
- Exemple:

```
ENTRA EL NIVELL:
2
ENTRA EL CARACTER:
#
FRACTAL:
####
# #
##
#
```

ACME, exercici 4 (triangle de Sierpinski)

- Exemples:

```
ENTRA EL NIVELL:
3
ENTRA EL CARACTER:
*
FRACTAL:
********
* * * *
**  **
*   *
****
* *
**
*


ENTRA EL NIVELL
1
ENTRA EL CARACTER
&
FRACTAL
&&
&
```

METODOLOGIA I TECNOLOGIA DE LA **PROGRAMACIÓ 2**

Sessió 11: Recursivitat

Anton Bardera, Miquel Feixas, Jaume Rigau,
Joan Surrell, Màrius Vila, Pau Xiberta

Curs 2019-20

Àrea LSI, Departament IMAE, Universitat de Girona