

METODOLOGIA I TECNOLOGIA DE LA **PROGRAMACIÓ 1** (Graus GEINF, GDDV i GEB)

Esteve del Acebo, Francesc Castro, Miquel
Feixas, Marta Fort, Jaume Rigau, Mateu Villaret

Curs 2019-20

Àrea LSI, Departament IMAE, Universitat de Girona

Índex

- S1. Introducció a l'entorn de programació
- S2. Tipus elementals de dades i instruccions bàsiques
- S3. Decisions
- S4. Bucles
- S5. Accions i funcions
- S6. Disseny descendent i tuples
- S7. Seqüències - I
- S8. Seqüències - II
- S9. Taules - I
- S10. Taules - II**
- S11. Taules i tuples
- S12. Pràctica final

METODOLOGIA I TECNOLOGIA DE LA PROGRAMACIÓ 1

S10: Taules - II

Esteve del Acebo, Francesc Castro, Miquel
Feixas, Marta Fort, Jaume Rigau, Mateu Villaret

Curs 2019-20

Àrea LSI, Departament IMAE, Universitat de Girona

Plantejament

- Repàs solució PAC1
- Matriu (taula bidimensional)
- Exercicis ACME

Matriu: definició i ús

- S'acostuma a **definir el tipus Matriu_tipus**

```
const int N_FIL_MAX=4, N_COL_MAX=6;  
// taules de tipus bàsics  
typedef int Matriu_enter[N_FIL_MAX][N_COL_MAX];  
typedef Vector_enter Matriu_enter[N_FIL_MAX]; // no l'usarem  
int main() {  
    Matriu_enter mat; //o be: int mat[4][6];
```

...

mat	0	1	2	3	4	5
0	-5	4	7	0	9	2
1	3	-23	4	-12	2	4
2	3	6	-8	0	1	2
3	12	11	5	13	7	9

mat[2] →

mat[3][1] →

PAS DE PARÀMETRES: Com a les taules unidimensionals

Pas de paràmetres

```
#include <iostream>
using namespace std;
const int N=10;
const int M=12;
typedef int MatriuEnter[N][M];

void llegirMatriu(MatriuEnter t){ // t és de sortida
//Pre: --
//Post: t conté N*M valors entrats
    for(int i=0; i<N; i++)
        for(int j=0; j<M; j++)
            cin>>t[i][j];
}

void mostrarMatriu(const MatriuEnter t){ // t és d'entrada
//Pre: --
//Post: s'han mostrat els N*M valors de t
    for(int i=0;i<N; i++){
        for(int j=0; j<M; j++)
            cout<<t[i][j]<<" ";
        cout<<endl; // salt de línia al final de cada fila
    }
}

...
```

Recorreguts en Matrius

Programa 1

/* Entrada: una matriu d'enters entrada per files i l'opció mínim o màxim
Sortida: Trobar, guardar i mostrar el mínim de cada fila o el màxim de cada columna */

```
...  
void minim_fila_matriu(const Matriu_enter mat, int n, int m, Vector_enter vec) {  
    // Pre: 0<=n<=N, 0<=m<=M  
    // Post: vec[0..n-1] conté el mínim de cada fila de mat[0..n-1,0..m-1]  
    for (int i=0; i<n; i++) {  
        int min=mat[i][0];  
        for (int j=1; j<m; j++)  
            if (mat[i][j]<min) min=mat[i][j];  
        vec[i]=min;  
    }  
}  
void maxim_columna_matriu(const Matriu_enter mat, int n, int m, Vector_enter vec) {  
    // Pre: 0<=n<=N, 0<=m<=M  
    // Post: vec[0..m-1] conté el màxim de cada columna de mat[0..n-1,0..m-1]  
    for (int j=0; j<m; j++) {  
        int max=mat[0][j];  
        for (int i=1; i<n; i++)  
            if (mat[i][j]>max) max=mat[i][j];  
        vec[j]=max;  
    }  
}  
...
```

- Llegim una matriu de 3x4, preguntem mínim o màxim i mostrem.
- Podem usar la funció mínim d'un vector? Com?

Cerca en Matrius

```
/* Entrada: 10x12 enters, identificador d'una fila
Sortida: Indiquem si la fila triada té o no negatius */
```

```
#include <iostream>
using namespace std;
const int N=10;
const int M=12;
typedef int MatriuEnter[N][M];

...
bool conteNegatiuFila(const MatriuEnter t, int n){
/* Pre: n>=0 i n<N
Post: retorna cert si t conté algun valor negatiu
a la fila n, i fals en cas contrari */
    bool trobat=false;
    int j=0;
    while(j<M and not trobat){
        if (t[n][j]<0) trobat=true;
        else j++;
    }
    return trobat;
}

...
```

```
...
int main()
{
    // Declaració de variables
    MatriuEnter t;
    int f;

    // llegim els elements de la matriu
    cout<<"ENTRA ELS ELEMENTS PER FILES"<<endl;
    llegirMatriu(t);

    cout<<"MATRIU ENTRADA "<<endl;
    mostrarMatriu(t);

    cout<<"FILA D'INTERES de 1 a "<<N<<endl;
    cin>>f; // compte amb la indexació !

    if (conteNegatiuFila(t,f-1))
        cout<<"LA FILA "<<f<<" CONTE NEGATIUS"<<endl;
    else
        cout<<"LA FILA "<<f<<" NO CONTE NEGATIUS"<<endl;
    return 0;
}
```


El tipus taula

La sintaxi següent: *tipus_base nom[]* estableix el tipus d'un array, però no necessàriament la capacitat. Exemple:

```
string llista_noms[] = {"pere", "anna", "maria"};
// llista_noms és una variable de tipus taula unidimensional
// d'string amb capacitat 3
```

Aquesta notació és especialment usada per declarar el tipus de paràmetres array. Exemple:

```
void copia_taula(const string original[], string copia[], int n) {
// Pre: n>=0 i n<= capacitat d'ambdues taules
// Post: copia conté els mateixos valors que original a 0..n-1
    for (int k=0; k<n; k++)
        copia[k]=original[k];
}
```

METODOLOGIA I TECNOLOGIA DE LA **PROGRAMACIÓ 1**

ACMEs

Exercicis ACME

Veure enunciat i detall entrada i sortida
ACME-style
al link de la sessió ACME dins “Moodle”



Errors permesos: **4**

Cost següents errors: **0.5** punts sobre 10

Temps: **17** dies

Exercicis ACME

ACME 1

Primers en una columna

Donada una matriu i un número de columna cal dir quants primers te la columna

ACME 2

Múltiple a la matriu

Donada una matriu i un nombre, trobar el primer múltiple del nombre donat a la matriu

ACME 3

Creació d'un quadrat màgic

Matriu quadrada a la qual files, columnes i diagonals sumen el mateix