# Seminar 8
# Matrice – part. I

**1. Sa se realizeze programul care calculeaza suma elementelor unei matrice dreptunghiulare de dimensiune m x n - A(mxn). Elementele matricei se vor citi de a tastatura.**

**Indicatie:**

Fie matricea $A = \begin{matrix} a(1,1) & a(1,2) & ... & a(1,n) \\ a(2,1) & a(2,2) & ... & a(2,n) \\ ... & ... & ... & ... \\ a(m,1) & a(m,2) & ... & a(m,n) \end{matrix}$.

Suma elementelor este:

a) S=a(1,1) + a(1,2) + ... +a(1,n) + a(2,1) + a(2,2) + ... + a)2,n) + .... + a(m,1) + ... + a(m,n) daca se insumeaza pe linii (lexicografic)
b) Sau S=a(1,1)+a(2,1)+...+a(m,1)+a(1,2)+...+a(m,2)+...+a(1,n)+ ... +a(m,n) daca se insumeaza pe coloane (invers lexicografic).

Matricea A poate fi privita ca un vector dimensiune mxn daca este liniarizat. (lexicografic sau invers lexicografic).

Algoritm:
S=0
- - - - - - - - - - - - -
i = 1
    j=1  S=S+a(1,1)=a(1,1)
    j=2  S=S+a(1,2)=a(1,1)+a(1,2)
    ...
    j=n  S=S+a(1,n)=a(1,1)+...+a(1,n)
- - - - - - - - - - - - -
i=2
    j=1  S=S+a(2,1)=a(1,1)+...+a(2,1)
         ...
    j=n  S=S+a(2,n)=a(1,1)+...+a(2,n)
- - - - - - - - - - - - -
i=m
    j=1  S=S+a(m,1)=a(1,1)+...+a(m,1)
         ...
    j=n  S=S+a(m,n)=a(1,1)+...+a(m,n)

Algoritmul recursiv poate fi scris astfel:
- formula de start: S=0;
- formula recursiva: S=S+a(i,j); i=1,m, j=1,n.

Elementele matricei se introduc de la tastatura, element cu element , cu ajutorul a doua structuri DO-FOR imbricate.
Obs: In C indicii incep de la 0.

**Exemplu**
Dati numarul de linii: 2
Dati numarul de coloane: 3

Dati elementele matricei:
a[1][1]=3
a[1][2]=2
a[1][3]=1
a[2][1]=5
a[2][2]=6
a[2][3]=8

3 2 1
5 6 8

Suma este: 25

**Rezolvare:**

```
INTREG n,m,i,j,s, a[100][100];
SCRIE(“Nr. de linii si de coloane:”);
CITESTE (m,n) ;

//citire matrice
DO-FOR i = 0,m-1,1
        DO-FOR j=0,n-1,1
                CITESTE (a[i][j]) ;
        ENDDO ;
ENDDO ;

//afisare matrice
DO-FOR i = 0,m-1,1
        DO-FOR j=0,n-1,1
                SCRIE (a[i][j]) ;
        ENDDO ;
ENDDO ;

//calcul suma
s=0 ;
DO-FOR i = 0,m-1,1
        DO-FOR j=0,n-1,1
                s= s + a[i][j] ;
        ENDDO ;
ENDDO ;

SCRIE(“Suma este:”, s);
```

STOP.

**2.Sa se realizeze programul care determina elementele maxim si minim dintr-o matrice dreptunghiulara A(mxn).**

**Indicatie**

Fie matricea $A = \begin{matrix} a(1,1) & a(1,2) & ... & a(1,n) \\ a(2,1) & a(2,2) & ... & a(2,n) \\ ... & ... & ... & ... \\ a(m,1) & a(m,2) & ... & a(m,n) \end{matrix}$. Liniarizand matricea lexicografic se obtine un vector de dimensiune mxn: A=(a(1,1), a(1,2), …,a(1,n), a(2,1),…a(2,n),…,a(m,1),…,a(m,n). Elementele MAX si MIN se determina ca fiind valoare maxima si respectiv minima din vector.
Algoritmul recursiv poate fi descris astfel:
- formule de start: MAX=a(1,1); MIN=a(1,1);
- formule recursive : MAX=max{MAX,a(i,j)} ; i=1,m ; j=1,n ;
  MIN = min{MIN,a(i,j)}; i=1,m ; j=1,n ;

Daca matricea se considera liniarizata invers lexicografic algoritmul este similar, cu deosebirea ca ordinea de variatie a indicilor este inversa. Maximul si minimul se vor determina concomitent, pornind de la ipoteza ca un element oarecare a(i,j) se poate afla:

- in intervalul (MAX, +∞) si este noua valoare maxima;
- in intervalul (-∞, MIN) si este noua valoare minima;
- in intervalul [MIN, MAX], caz in care nu afecteaza niciuna din valorile cautate.

**Exemplu**
Dati numarul de linii: 2
Dati numarul de coloane: 3

Dati elementele matricei:
a[1][1]=3
a[1][2]=2
a[1][3]=1
a[2][1]=5
a[2][2]=6
a[2][3]=8

3 2 1
5 6 8

Max. este: 8
Min. este: 1

**Rezolvare**
INTREG n,m,i,j,max,min, a[100][100];

```
SCRIE(“Nr. de linii si de coloane:”);
CITESTE (m,n) ;

//citire matrice
DO-FOR i = 0,m-1,1
        DO-FOR j=0,n-1,1
                CITESTE (a[i][j]) ;
        ENDDO ;
ENDDO ;

//afisare matrice
DO-FOR i = 0,m-1,1
        DO-FOR j=0,n-1,1
                SCRIE (a[i][j]) ;
        ENDDO ;
ENDDO ;

//calcul max si min
max=a[0][0];
min=a[0][0];

DO-FOR i = 0,m-1,1
        DO-FOR j=0,n-1,1
                { IF (max<a[i][j]) THEN max=a[i][j]; ENDIF;
                  IF (min>a[i][j]) THEN  min=a[i][j]; ENDIF;
                 }
        ENDDO ;
ENDDO ;

SCRIE(“Max. este:”, max);
SCRIE(“Min. este:”, min) ;

STOP.
```

**3. Sa se scrie programul care realizeaza determinarea pozitiei primei aparitii a unei valori date intr-o matrice dreptunghiulara A(mxn).**

**Indicatii**

- Matricea se va parcurge lexicografic.
- Cand se identifica valoarea cautata se afiseaza pozitia (linia si coloana) si se abandoneaza parcurgerea matricei. Iesirea fortata din structurile repetitive se realizeza cu ajutorul unei variabile care ia valoarea 1 daca valoarea data a fost regasita printre elementele matrice sau 0 in caz contrar. Testarea suplimentara a variabilei, alaturi de conditia de sfarsit de linii (i>m) si sfarsit de coloane (j>n), transforma structurile repetitive cu numarator in doua structuri WHILE-DO imbricate.
- In final, daca variabila este egala cu 0, inseamna ca valoarea nu a fost regasita si se afiseaza un mesaj corespunzator

**Obs: In C, indicii pleaca de la 0.**
**Exemplu**
Dati numarul de linii: 2
Dati numarul de coloane: 3

Dati elementele matricei:
a[1][1]=3
a[1][2]=2
a[1][3]=1
a[2][1]=5
a[2][2]=6
a[2][3]=8

Valoarea cautata este: 2

3 **2** 1
5 6 8

Valoarea cautata se afla pe linia 0 si coloana 1

**Rezolvare**

```
INTREG n,m,i,j,pozl, pozc, gas, val, a[100][100];
SCRIE("Nr. de linii si de coloane:");
CITESTE (m,n) ;

//citire matrice
DO-FOR i = 0,m-1,1
        DO-FOR j=0,n-1,1
                CITESTE (a[i][j]) ;
        ENDDO ;
ENDDO ;

CITESTE (val);

//afisare matrice
DO-FOR i = 0,m-1,1
        DO-FOR j=0,n-1,1
                SCRIE (a[i][j]) ;
        ENDDO ;
ENDDO ;

//identificare valoare
gas=0;
i=0;
WHILE ((i<m) &&(!gas)) DO
        {       j=0;
                WHILE ((j<n) && (!gas)) DO
                        {       IF (a[i][j]= =val) THEN
```

```
                                                    { gas = 1;
                                                      pozl = i;
                                                      pozc = j;
                                                    }
                                ENDIF;
                               j=j+1;
                        }
                ENDWHILE;
                i=i+1;
        }
ENDWHILE;

IF (gas) THEN SCRIE (“Val. cautata se afla pe linia si coloana:”, pozl,pozc);
ENDIF;

STOP.
```

**4. Sa se scrie programul care sorteaza elementele unei matrice A(mxn) in ordine crescatoare.**

**Indicatie**
Se va apela la un artificiu de programare – copierea tuturor elementelor din matrice intr-un vector. Acest vector este sortat si ulterior, elementele sortate sunt redistribuite in matrice.

**Rezolvare**

```
INTREG n,m,i,j,k, l, a[10][10], c[10][10], v[100], aux;
SCRIE(“Nr. de linii si de coloane:”);
CITESTE (n,m) ;

//citire matrice
DO-FOR i = 0,n-1,1
        DO-FOR j=0,m-1,1
                CITESTE (a[i][j]) ;
        ENDDO ;
ENDDO ;


//afisare matrice
DO-FOR i = 0,n-1,1
        DO-FOR j=0,m-1,1
                SCRIE (a[i][j]) ;
        ENDDO ;
ENDDO ;

//sortare
k=0;
DO-FOR i = 0,n-1,1
```

```
        DO-FOR j=0,m-1,1
                {       v[k]=a[i][j];
                        k=k+1;
                }
        ENDDO ;
ENDDO ;

DO-FOR i = 0,k-2,1
        DO-FOR j=i+1,k,1
                IF (v[i]>v[j])  {       aux= v[i];
                                        v[i]=v[j];
                                        v[j]=aux;
                                }
                ENDIF;
        ENDDO ;
ENDDO ;

DO-FOR i = 0,k-1,1 SCRIE (v[i]);
ENDDO ;

l=0;
WHILE (l<n*m)
        {
                i=0;
                WHILE (i<n)
                        {  j=0;
                           WHILE  (j<m)
                                { c[i][j]= v[l];
                                  j=j+1;
                                  l=l+1;
                                }
                           ENDWHILE;
                           i=i+1;
                        }
                 ENDWHILE;
}
 ENDWHILE;

DO-FOR i = 0,n-1,1
        {  DO-FOR j=0,m-1,1
                SCRIE (c[i][j]) ;
           ENDDO ;
        }
ENDDO;

STOP.
```