Tablouri bidimensionale

Prelucrări asupra tablourilor bideminsionale (matricelor)

a) **Definiţie**

Tablou bidimensional = succesiune de locaţii de memorie recunoscute prin acelaşi identificator **şi** prin poziţia fiecăreia în cadrul şirului. Poziţia este dată printr-o suită de două numere pozitive (indecşi), care reprezintă cele două dimensiuni (linie şi coloană).

Exemplu:

T	0	1	2	3	4
0					
1					
2					
3					

Tabloul T

Are 4 linii(linii orizontale) numerotate de la 0 la 3

Are 5 coloane(linii verticale) numerotate de la 0 la 4

Valorile elementelor tabloului trebuie sa aiba acelasi tip

Declarare tablou**: int T[4][5]**

In memoria calculatorului elementele tabloului sunt memorate astfel:

T	0	1	2	3	4
0	0	12		2	
1	8	11	8	6	
2					
3					

T[i][j]=elementul care se afla la intersectia liniei i cu coloana j; i este cuprins intre 0 si 3 , iar j este cuprins intre 0 si 4;

Dimensiunea unei matrici(tablou dimensional) este nxm, unde m este numarul de linii, iar n este numarul de coloane.

T	**0**	**1**	**2**	**3**	**4**	.....	n-2	n-1
0	0	12		2		.....		
1	8	11	8	6		.....		
2						.....		
3						.....		
.....	.....	.....	.....	.....	.....	.....	.....	.....
m-1						.....		
m-2						.....		

Pentru a lucra cu tablouri trebuie mai intai sa dom valori elementelor matricii. Trebuie sa declaram o matrice cu 100 de linii sau 100 de coloane(sau alte numere).Vom lucra cu m linii si n coloane.
Putem sa lucram cu liniile 0,1,...,n-1 si cu coloanele 0,1,...,n-1 sau putem sa lucram cu liniile 1,2,...,m si cu coloanele 1,2,...,n.
Putem sa dam valori pe LINII:

```
#include<iostream.h>
int main(void)
{
	a[100][100];
	int n, m, i, j;
	cout<<"Dati dimensiunile matricei A"<<endl;
	cout<<"Dati numarul de linii m = "; cin>>m;
	cout<<"Dati numarul de coloane n = "; cin>>n;

	for(i=0; i<m; i++)
	  for(j=0; j<n; j++)
	  {
		cout<<"a["<<i<<", "<<j<<"] = "; cin>>a[i][j];
	  }
}
```

sau

```
#include<iostream.h>
int main(void)
{
        a[100][100];
        int n, m, i, j;
        cout<<"Dati dimensiunile matricei A"<<endl;
        cout<<"Dati numarul de linii m = "; cin>>m;
        cout<<"Dati numarul de coloane n = "; cin>>n;

        for(i=1; i<=m; i++)
          for(j=1; j<=n; j++)
          {
                  cout<<"a["<<i<<", "<<j<<"] = "; cin>>a[i][j];
          }
}
```

Putem sa dam valori pe COLOANE:

```
#include<iostream.h>
int main(void)
{
        a[100][100];
        int n, m, i, j;
        cout<<"Dati dimensiunile matricei A"<<endl;
        cout<<"Dati numarul de linii m = "; cin>>m;
        cout<<"Dati numarul de coloane n = "; cin>>n;

        for(i=0; i<n; i++)
          for(j=0; j<m; j++)
          {
                  cout<<"a["<<i<<", "<<j<<"] = "; cin>>a[i][j];
          }
}
```

sau

```
#include<iostream.h>
int main(void)
{
	a[100][100];
	int n, m, i, j;
	cout<<"Dati dimensiunile matricei A"<<endl;
	cout<<"Dati numarul de linii m = "; cin>>m;
	cout<<"Dati numarul de coloane n = "; cin>>n;

	for(i=1; i<=n; i++)
	  for(j=1; j<=m; j++)
	  {
		cout<<"a["<<i<<", "<<j<<"] = "; cin>>a[i][j];
	  }
}
```

Scrierea unei matrici pe ecran, astfel incat liniile sa fie unele sub altele, iar elementele de pe o linie sa fie separate prin cate un spatiu.

```
#include<iostream.h>
int main(void)
{
	a[100][100];
	int n, m, i, j;
	cout<<"Dati dimensiunile matricei A"<<endl;
	cout<<"Dati numarul de linii m = "; cin>>m;
	cout<<"Dati numarul de coloane n = "; cin>>n;

	for(i=1; i<=n; i++)
	  for(j=1; j<=m; j++)
	  {
		cout<<"a["<<i<<", "<<j<<"] = "; cin>>a[i][j];
	  }


	for(i=1; i<=n; i++)
	  {for(j=1; j<=m; j++)
		cout<<a[i][j]<<' ';
		cout<<endl;
	  }


}
```

MARGINE MATRICE=toate elementele de pe margine(prima linie, prima coloana, ultima linie, ultima coloana)

Matrice patratica

Într-o matrice pătratică numarul de linii= numarul de coloane (n=m).

Într-o matrice pătratică avem:

- Diagonala principala elementele a[i][i], cu i=1,n sau a[i][i], cu i=0,n-1
- Diagonala secundara elementele a[i][n-i+1], i=1,n sau a[i][n-i-1], i=0,n-1

Zonele determinate de diagonale:

I.

Pe diagonala principală i=j

Sub diagonala principala: i>j

Deasupra diagonalei principale: i<j

II.

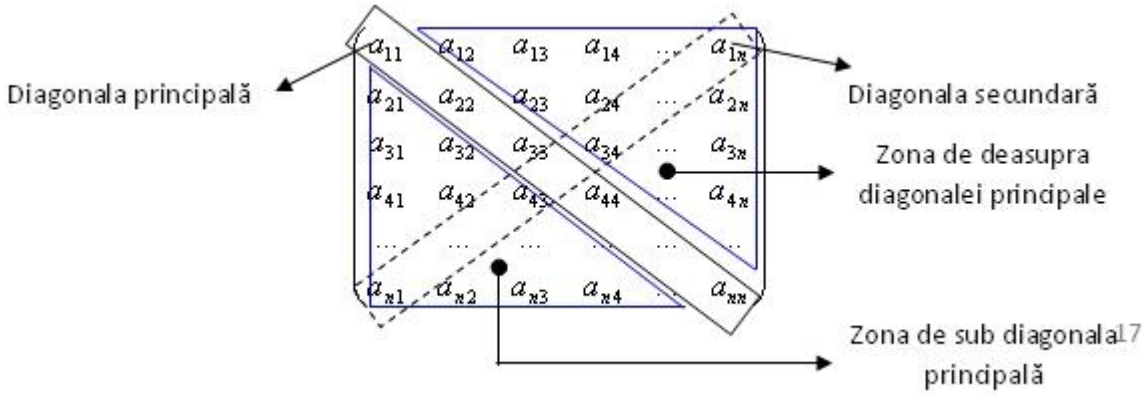
Pe diagonala secundară j=n-i+1

Sub diagonala secundara: j>n-i+1

Deasupra diagonalei secundare:j<n-i+1

ZONE speciale in matrice patratice

1. Diagonala principala si secundara



Diagonala principală

Diagonala principală este formată din elementele care îndeplinesc relaţia

i = j – numărul liniei este egal cu numărul coloanei pe care se află.

Diagonala secundară

Diagonala secundară conţine elementele a1n, a2 n-1 , a3 n-2,…,an1 caracterizate de relaţia

i+j = n+1.

Zona de deasupra diagonalei principale

Elementele de deasupra diagonalei principale sunt a12, a13, a14,…,a1n, a23, a24, a25,…,a2n, …., an-1 n-1, an-1 n.

Relaţia dintre coordonate comună tuturor elementelor din această zonă este **i < j** .

Zona de sub diagonala principală
Elementele a21, a31, a32,…,a41, a42, a43, …..,an1, an2, an n-1 se află sub diagonala principală şi au între coordonate relaţia **i > j**.
În practică prelucrarea elementelor se poate face exclusiv pe diagonale respectiv pe zonele identificate mai sus(ex:ordonarea diagonalelor respectiv verificarea simetriei sau a triunghiularităţii) sau se poate opta pentruo parcurgere a tuturor elementelor matricei şi prelucrarea diferenţiată a elementelor în funcţie de relaţia dintre coordonate(ex: completarea elementelor cu anumite valori, calculul simultan al mai multor rezultate obţinute pentru fiecare zonă în parte).

Modalităţi de prelucrare a elementelor în matrice pătratică de dimensiune n
Diagonala principala:
for (i=1;i<=n;i++)
<prelucrează a[i][i]>
Diagonala secundara:
for (i=1;i<=n;i++)
<prelucrează a[i][n-i+1]>
Deasupra diagonalei principale:
for (i=1;i<=n-1;i++)
for(j=i+1;j<=n;j++)
<prelucrează a[i][j]>
Sub diagonala principala:
for (i=2;i<=n;i++)
for(j=1;j<=i-1;j++)
<prelucrează a[i][j]>
Prelucrarea intr-o singura parcurgere a tuturor zonelor:
for (i=1;i<=n;i++)
for(j=1;j<=n;j++)
if (i==j)*<prelucrează a[i,j] – diag. princ.>
else
if (i+j==n+1)
*<prelucrează a[i,j] – diag. sec.>
else
if (i>j)
*<prelucrează a[i,j] – deasupra diag. princ.>
else
*<prelucrează a[i,j] – sub diag. princ.>
if (i+j<n+1)
*<prelucrează a[i,j] – deasupra diag. sec.>
if (i+j>n+1)
*<prelucrează a[i,j] – sub diag. sec.>

d) **Prelucrări asupra matricelor**

Ex. Pentru o matrice dată să se calculeze suma elementelor care aparţin unui interval dat (x_{inf}<=tab[i][j] && x_{sup}>=tab[i][j]).
Prezentarea algoritmului :

- se citesc capetele intervalului în care trebuie să se încadreze elementele cautate în matrice
  x_{inf} şi x_{sup}
- se citesc dimensiunile matricei
  m şi n
- se citesc elementele matricei

```
pentru i=0,m-1 execută
        pentru j=0,n-1 execută
                citeşte tab[i][j];
        sfârşit pentru
sfârşit pentru
```

- *suma = 0;*
- se parcurge matricea element cu element. Se testează dacă elementul curent se încadrează în intervalul dorit şi în caz afirmativ elementul curent se adună la suma calculată anterior

```
pentru i=0,m-1 execută
        pentru j=0,n-1 execută
                dacă tab[i][j] > xi şi  tab[i][j] < xf atunci
                        suma  = suma + tab[i][j];
                sfârşit dacă
        sfârşit pentru
sfârşit pentru
```

- afişează suma

Obs. Variabila **suma** reprezintă suma calculată.

Ex. 7 Să se determine elementul maxim de pe fiecare linie dintr-o matrice

```
#include <stdio.h>
#include <conio.h>

void main()
{
int tab[10][10],max_lin[10];
int i,j,m,n,max;

clrscr();
printf("\n Introduceti dimensiunile matricei ");
scanf("%d %d",&m,&n);
printf("\n Introduceti elementele matricei\n");

for(i=0;i<m;i++)
{
        for(j=0;j<n;j++)
                {
                        printf("tab[%d][%d]=",i,j);
                        scanf("%d",&tab[i][j]);
                }
}

printf("\n Matricea citita de la tastatura este: \n");
for(i=0;i<m;i++)
{
printf("\t");
        for(j=0;j<n;j++)
                printf(" %4d",tab[i][j]);
printf("\n");
}

for(i=0;i<m;i++)
```

```
{
max=matr[i][0];
        for(j=0;j<n;j++)
        {
          if(max < tab[i][j])
                        max=tab[i][j];
        }
max_lin[i]=max;
}
printf("\n elementele vectorului de maxime sunt");
for(i=0;i<m;i++)
        printf(" %5d",max_lin[i]);
getch();
}
```

Obs. Vectorul **max_lin** conţine elementele maxime de pe fiecare linii. Evident acesta are dimensiunea egală cu numărul liniilor matrucii tab.

e) Operaţii din algebra matriceală

Ex. 8 Să se determine matricea transpusă a unei matrice citită de la tastatură **(matr_A)**

Prezentarea algoritmului :

- se citesc dimensiunile matricelor
  m şi n
- se citesc elementele matricei iniţiale
  pentru i=0,m-1 execută
  pentru j=0,n-1 execută
  citeşte matr_A[i][j];
  sfârşit pentru
  sfârşit pentru
- ecou
- se parcurge matricea iniţială element cu element, elementul cu indicele [i][j] devenind elementul cu indicele [j][i] în matricea finală
  pentru i=0,m-1 execută
  pentru j=0,n-1 execută
  matr_B[i][j] = matr_A[j][i]
  sfârşit pentru
  sfârşit pentru

- se afişează elementele matricei finale (matricea transpusă)
  pentru i=0,n-1 execută
  pentru j=0,m-1 execută
  afişează matr_B[i][j];
  sfârşit pentru
  sfârşit pentru

Ex. Să se determine matricea produs rezultată în urma înmulţirii a doua matrice:
matr_C(m,n) = matr_A(m,n) x matr_B(m,n)

Obs. Programul nu verifică corectitudinea dimensiunilor matricelor pentru a efectua operaţia de înmulţire.

```
#include <stdio.h>
#include <conio.h>
void main()
{
int matr_A[10][10], matr_B[10][10];
int matr_C[10][10];
int i,j,k,m,n,p;
clrscr();
printf("\n Introduceti dimensiunile matricelor ");
scanf("%d %d %d",&m,&n,&p);
printf("\n Introduceti elementele primei matrici \n");
for(i=0;i<m;i++)
        for(j=0;j<n;j++)
        {
                printf("matr_A[%d][%d]=",i,j);
                scanf("%d",&matr_A[i][j]);
         }

printf("\n Matricea citita de la tastatura este: \n");
for(i=0;i<m;i++)
{
printf("\t");
        for(j=0;j<n;j++)
                printf(" %4d",matr_A[i][j]);
printf("\n");
}

printf("\n Introduceti elementele pentru matr B \n");
for(i=0;i<n;i++)
        for(j=0;j<p;j++)
        {
                printf("matr_B[%d][%d]=",i,j);
                scanf("%d",&matr_B[i][j]);
         }

printf("\n Matricea citita de la tastatura este: \n");
for(i=0;i<n;i++)
{
printf("\t");
        for(j=0;j<p;j++)
                printf(" %4d",matr_B[i][j]);
printf("\n");
}

for(i=0;i<m;i++)
        for(j=0;j<p;j++)
                  matr_C[j][i] = 0;

for(i=0;i<m;i++)
        for(j=0;j<p;j++)
                for(k=0;k<n;k++)
    matr_C[i][j] = matr_C[i][j]+matr_A[i][k]*matr_B[k][j];

printf("\n Matricea produs \n");
for(i=0;i<m;i++)
{
printf("\t");
        for(j=0;j<p;j++)
                printf(" %4d",matr_C[i][j]);
printf("\n");
}
getch();
```