

Programarea calculatoarelor

8 C++ Tablouri

Adrian Runceanu
www.runceanu.ro/adrian

Curs 8

Tablouri

Capitolul 7. Tablouri

7.1. Generalități. Clasificare

7.2. Tablouri unidimensionale (vectori)

7.3. Tablouri bidimensionale (matrici)

7.4. Tablouri multidimensionale

7.1. Generalitati. Clasificare

Numim **tablou** *o colecție de date de același tip, în care elementele sunt ordonate, iar accesul la fiecare element are loc prin indice.*

În funcție de numărul indicilor avem mai multe tipuri de tablouri:

1. Tablouri unidimensionale (cu un singur indice)
2. Tablouri bidimensionale (cu doi indici)
3. Tablouri multidimensionale (cu mai mulți indici)

Capitolul 7. Tablouri

7.1. Generalități. Clasificare

7.2. Tablouri unidimensionale (vectori)

7.3. Tablouri bidimensionale (matrici)

7.4. Tablouri multidimensionale

7.2. Tablouri unidimensionale (vectori)

- Tablourile unidimensionale funcționează ca un vector și se pot declara astfel:

```
tip nume_tablou[dimensiune_maximă];
```

- Se observă că este **obligatorie folosirea parantezelor drepte** care să încadreze dimensiunea maximă pe care o alege utilizatorul pentru acel tablou unidimensional.

7.2. Tablouri unidimensionale (vectori)

Exemplu:

Declarări de tablouri unidimensionale:

```
int a[25];    // declararea unui tablou unidimensional cu maxim 25 de  
              // elemente, fiecare de tip întreg  
float x[30]; // declararea unui tablou unidimensional cu maxim 30 de  
              // elemente, fiecare de tip real simplă precizie  
char s[40];  // declararea unui tablou unidimensional cu maxim 40 de  
              // elemente, fiecare de tip caracter
```

7.2. Tablouri unidimensionale (vectori)

- Compilatorul C++ *alocă un spațiu de memorie egal cu numărul maxim de elemente ale tabloului*, rezervând bineînțeles octeți în funcție de tipul de bază al fiecărui tablou.
- *Accesul la fiecare element al tabloului se face prin numele acestuia urmat între paranteze, de indicele său (adică poziția pe care o ocupă în tablou).*
- În limbajul C++, *indicii tablourilor încep numărătoarea de la valoarea 0.*

7.2. Tablouri unidimensionale (vectori)

Exemplu:

Modalități de acces la elementele ce pot fi memorate în tablourile unidimensionale declarate anterior:

$a[0]$ – reprezintă elementul aflat pe prima poziție în tablou

$a[24]$ - reprezintă elementul aflat pe ultima poziție în tablou

$x[i]$ - *reprezintă elementul aflat pe poziția i în tablou*, unde i poate avea valori între 0 și 29.

7.2. Tablouri unidimensionale (vectori)

Inițializarea elementelor unui tablou se poate face total sau parțial la declararea lor:

```
int b[5] = {1, 2, 3, 4, 5};
```

Astfel:

- elementul b[0] are valoarea 1
- elementul b[1] are valoarea 2
- elementul b[2] are valoarea 3
- elementul b[3] are valoarea 4
- elementul b[4] are valoarea 5

7.2. Tablouri unidimensionale (vectori)

Problema 1:

Se consideră n numere reale. Se cere să se determine valoarea minimă și valoarea maximă.

Exemplu:

Date de intrare:

$n=5$ și $x=\{10, -2, 34, -198, 4\}$

Date de ieșire:

minim=-198, maxim=34

7.2. Tablouri unidimensionale (vectori)

```
#include<iostream.h>
```

```
int main(void)
```

```
{
```

```
    int i, n;
```

```
    float x[50], min, max;
```

```
    cout<<"Dati numarul de elemente ale tabloului ";
```

```
    cin>>n;
```

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

```
    {
```

```
        cout<<"x["<<i+1<<"]="<< ";
```

```
        cin>>x[i];
```

```
    }
```

Citirea numarului de
elemente ce vor fi
prelucrate in vector - n

Citirea elementelor si
memorarea lor in
vectorul x

7.2. Tablouri unidimensionale (vectori)

```
min = x[0];      max = x[0];  
for( i=1; i<n; i++ )  
    if( min > x[i] )  
        min = x[i];  
    else  
        if( max < x[i] )  
            max = x[i];  
  
cout<<"\nMinimul este "<<min;  
cout<<"\nMaximul este "<<max;  
}
```

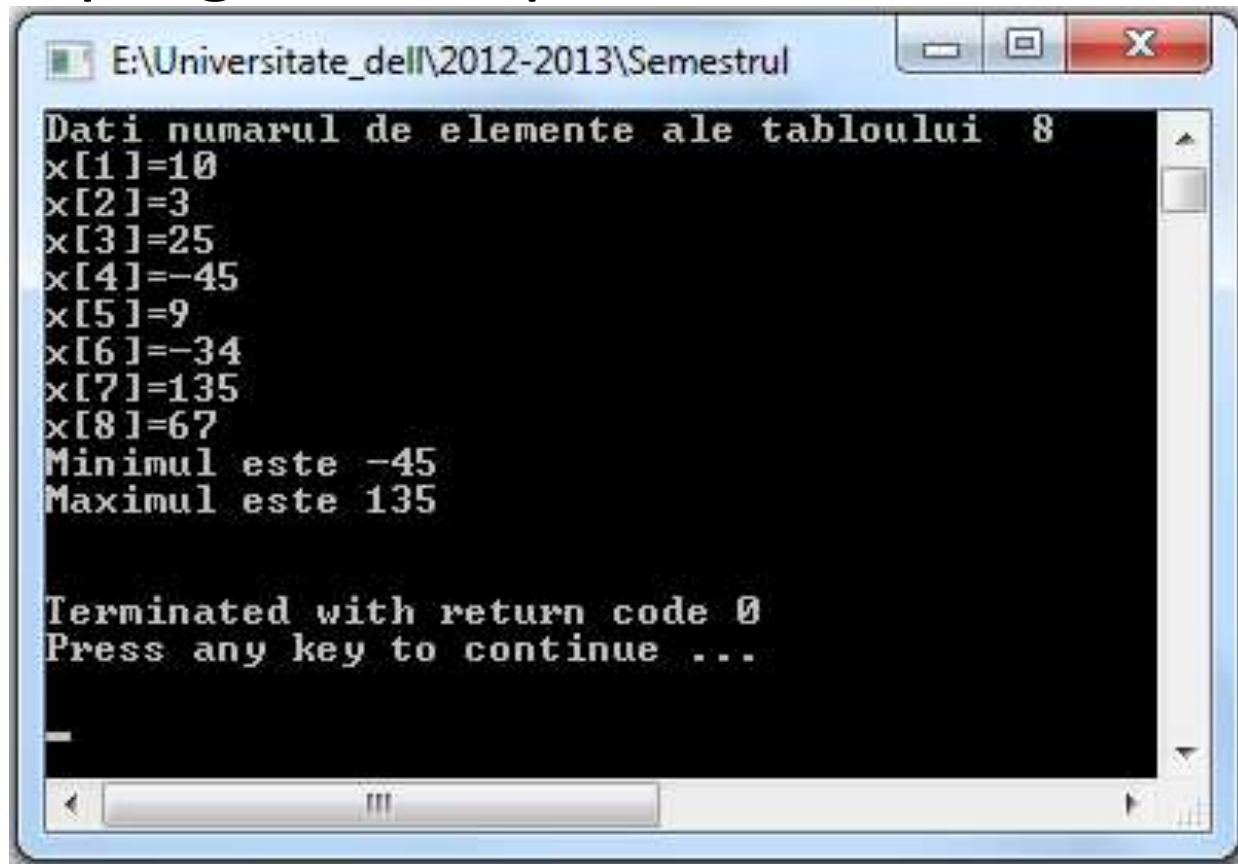


Determinarea
minimului

Determinarea
maximului

7.2. Tablouri unidimensionale (vectori)

Executia programului pe o serie de date de test:



```
E:\Universitate_dell\2012-2013\Semestrul
Dati numarul de elemente ale tabloului 8
x[1]=10
x[2]=3
x[3]=25
x[4]=-45
x[5]=9
x[6]=-34
x[7]=135
x[8]=67
Minimul este -45
Maximul este 135

Terminated with return code 0
Press any key to continue ...
```

7.2. Tablouri unidimensionale (vectori)

Problema 2:

Se consideră n numere întregi.

Se cere să se verifice dacă ele sunt sau nu în ordine crescătoare, afișând câte un mesaj corespunzător.

Exemplu:

Date de intrare:

$n=5$ și $x=\{1, 2, 17, 25, 43\}$

Date de ieșire:

Elemente vectorului sunt în ordine crescătoare

7.2. Tablouri unidimensionale (vectori)

```
#include<iostream.h>
```

```
int main(void)
```

```
{
```

```
    int i, n, verific=1;
```

```
    int x[50];
```

```
    cout<<"Dati numarul de elemente ale tabloului X ";
```

```
    cin>>n;
```

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

```
    {
```

```
        cout<<"x["<<i+1<<"]= ";
```

```
        cin>>x[i];
```

```
    }
```

Citirea numarului de
elemente ce vor fi
prelucrate in vector - n

Citirea elementelor si
memorarea lor in
vectorul x

7.2. Tablouri unidimensionale (vectori)

Verificarea
proprietatii cerute
in enunt

```
for(i=0; i<=n-2; i++)  
    if( x[i] > x[i+1] ) verific=0;
```

```
if( verific == 1 ) cout<<"Numerele din tablou sunt in  
ordine CRESCATOARE";  
else cout<<"Numerele din tablou NU sunt in ordine  
CRESCATOARE";  
}
```

E:\Universitate_de\2012-2013\Semestrul

Dati numarul de elemente ale tabloului X 7

x[1]= 2

x[2]= 24

x[3]= 39

x[4]= 42

x[5]= 53

x[6]= 78

x[7]= 123

Numerele din tablou sunt in ordine CRESCATOARE

E:\Universitate_de\2012-2013\Semestrul

Dati numarul de elemente ale tabloului X 5

x[1]= 1

x[2]= 2

x[3]= 3

x[4]= 5

x[5]= 4

Numerele din tablou NU sunt in ordine CRESCATOARE

Terminated with return code 0

Press any key to continue ...

Capitolul 7. Tablouri

7.1. Generalități. Clasificare

7.2. Tablouri unidimensionale (vectori)

7.3. Tablouri bidimensionale (matrici)

7.4. Tablouri multidimensionale

7.3. Tablouri bidimensionale (matrici)

- Tablourile bidimensionale funcționează ca o matrice și se pot declara astfel:

```
tip nume_tablou[dim_linie][dim_coloana];
```

- La fel ca și în cazul tablourilor unidimensionale, și în cazul tablourilor bidimensionale, la declarare, se trece **dimensiunea maximă a liniilor** (**dim_linie**) și **dimensiunea maximă a coloanelor** (**dim_coloana**).

7.3. Tablouri bidimensionale (matrici)

Exemplu:

Declarări de tablouri bidimensionale:

```
int a[10][10]; // declararea unui tablou bidimensional cu maxim 100 de  
                // elemente (10*10), fiecare de tip întreg  
float x[5][5]; // declararea unui tablou bidimensional cu maxim 25 de  
                // elemente(5*5), fiecare de tip real simplă precizie  
char s[20][10]; // declararea unui tablou bidimensional cu maxim 200 de  
                // elemente(20*10), fiecare de tip caracter
```

7.3. Tablouri bidimensionale (matrici)

- Compilatorul C++ *alocă un spațiu de memorie egal cu numărul de linii înmulțit cu numărul de coloane ale tabloului*, rezervând bineînțeles octeți în funcție de tipul de bază al fiecărui tablou.
- *Accesul la fiecare element al tabloului se face prin numele acestuia urmat între paranteze, de indicele liniei și indicele coloanei (adică poziția pe care o ocupă în tablou).*

7.3. Tablouri bidimensionale (matrici)

Exemplu:

Modalități de acces la elementele ce pot fi memorate în tablourile bidimensionale declarate anterior:

`a[0][0]` – reprezintă elementul aflat pe linia 0 coloana 0

`a[9][9]` - reprezintă elementul aflat pe linia 9 coloana 9

`x[i][j]` - *reprezintă elementul aflat pe linia i, coloana j în matrice*, unde i și j pot avea valori între 0 și 4.

7.3. Tablouri bidimensionale (matrici)

Inițializarea elementelor unui tablou *se poate face total sau parțial la declararea lor*:

```
int b[2][3]={ {1, 2, 3}, {4, 5, 6} };
```

Astfel:

- elementul b[0][0] are valoarea 1
- elementul b[0][1] are valoarea 2
- elementul b[0][2] are valoarea 3
- elementul b[1][0] are valoarea 4
- elementul b[1][1] are valoarea 5
- elementul b[1][2] are valoarea 6

7.3. Tablouri bidimensionale (matrici)

Problema 1:

Se consideră o matrice A cu $n \times m$ numere întregi. Se cere să se obțină **transpusa** sa.

Exemplu:

Date de intrare:

$n=3$, $m=4$ si matricea A :

1 2 3 4

5 6 7 8

9 10 11 12

Date de ieșire:

Matricea transpusa B :

$n=4$ si $m=3$

1 5 9

2 6 10

3 7 11

4 8 12

7.3. Tablouri bidimensionale (matrici)

```
#include<iostream.h>
```

```
int main(void)
```

```
{
```

```
    int a[10][10], b[10][10];
```

```
    int n, m, i, j;
```

```
    cout<<"Dati dimensiunile matricei A \n";
```

```
    cout<<"Dati numarul de linii n  = ";
```

```
    cin>>n;
```

```
    cout<<"Dati numarul de coloane m = ";
```

```
    cin>>m;
```

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

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


```
        {
```

```
            cout<<"a["<<i+1<<" , "<<j+1<<" ] = ";
```


```
            cin>>a[i][j];
```

```
        }
```

Citirea numarului de linii – n
si de coloane - m ale matricei




Citirea elementelor
si memorarea lor in
matricea a



7.3. Tablouri bidimensionale (matrici)

```
cout<<"Elementele matricei A sunt : \n";  
for(i=0; i<n; i++)  
{  
    for(j=0; j<m; j++) cout<<a[i][j]<<" ";  
    cout<<"\n";  
}
```



Afisarea elementelor
din matricea a

7.3. Tablouri bidimensionale (matrici)

```
for(i=0; i<n; i++)  
    for(j=0; j<m; j++)  
        b[j][i] = a[i][j];
```

Construirea matricei
transpose prin
transformarea liniilor in
coloane si invers

```
cout<<"Elementele matricei transpose sunt \n";
```

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

```
{
```

```
    for(j=0; j<n; j++) cout<<b[i][j]<<" ";
```

```
    cout<< "\n";
```

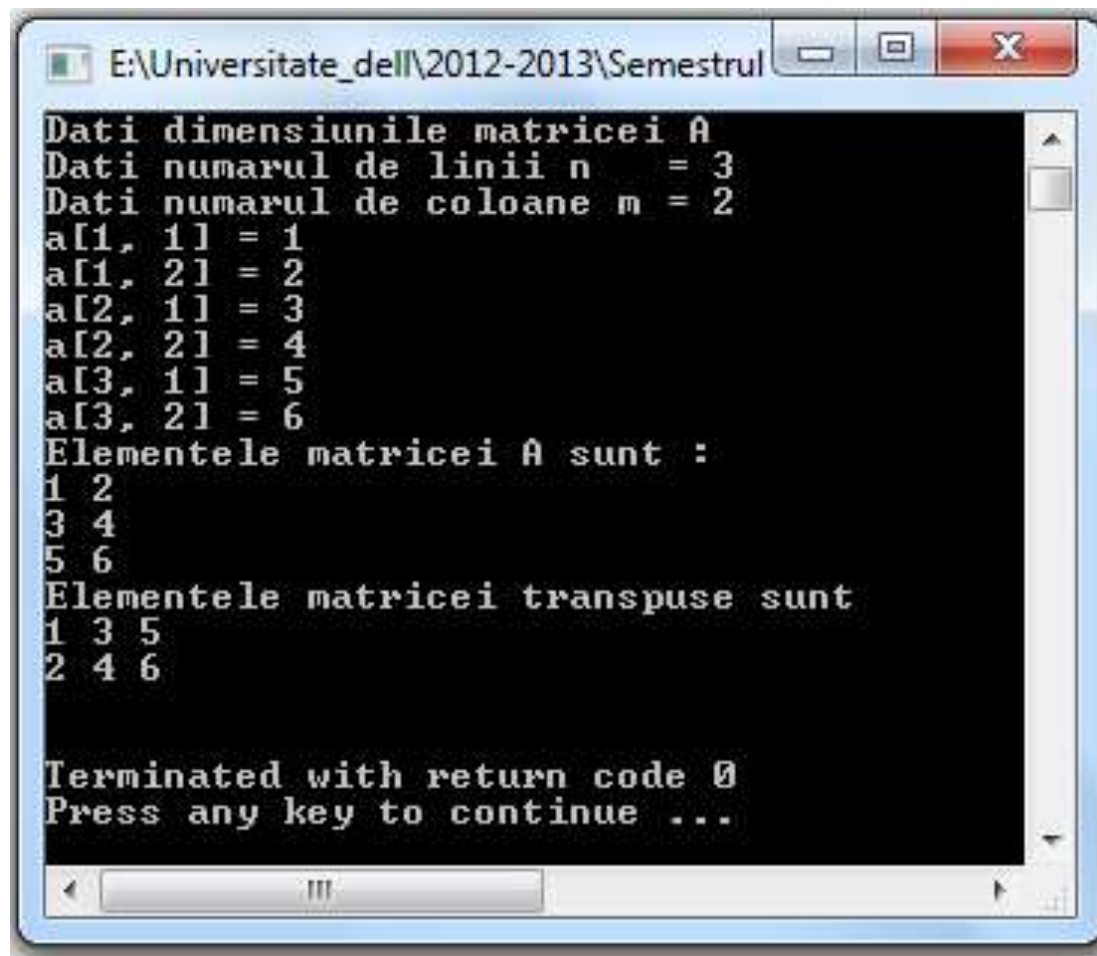
```
}
```

```
}
```

matrix	it's transpose								
<table><tr><td>a</td><td>-b</td></tr><tr><td>b</td><td>a</td></tr></table>	a	-b	b	a	<table><tr><td>a</td><td>b</td></tr><tr><td>-b</td><td>a</td></tr></table>	a	b	-b	a
a	-b								
b	a								
a	b								
-b	a								

7.3. Tablouri bidimensionale (matrici)

Executia programului pe o serie de date de test:



```
E:\Universitate_dell\2012-2013\Semestrul
Dati dimensiunile matricei A
Dati numarul de linii n = 3
Dati numarul de coloane m = 2
a[1, 1] = 1
a[1, 2] = 2
a[2, 1] = 3
a[2, 2] = 4
a[3, 1] = 5
a[3, 2] = 6
Elementele matricei A sunt :
1 2
3 4
5 6
Elementele matricei transpuse sunt
1 3 5
2 4 6

Terminated with return code 0
Press any key to continue ...
```

7.3. Tablouri bidimensionale (matrici)

Problema 2:

Se consideră două tablouri bidimensionale (matrici) A și B cu $n \times m$ numere întregi.

Se cere să se calculeze **matricea suma**:

$$C = A + B.$$

$$\begin{bmatrix} 3 & 8 \\ 4 & 6 \end{bmatrix} + \begin{bmatrix} 4 & 0 \\ 1 & -9 \end{bmatrix} = \begin{bmatrix} 7 & 8 \\ 5 & -3 \end{bmatrix}$$

7.3. Tablouri bidimensionale (matrici)

```
#include<iostream.h>
```

```
int main(void)
```

```
{
```

```
int Matrice1 [10][10], Matrice2 [10][10], Matricesuma[10][10];
```

```
int i, j, n, m;
```

```
cout<<"Dati dimensiunile primei matrici \n";
```

```
cout<<"Dati numarul de linii n  = ";      cin>>n;
```

```
cout<<"Dati numarul de coloane m = ";      cin>>m;
```

```
for(i=0; i<n; i++){
```


```
    for(j=0; j<m; j++){
```

```
        cout<<"Matrice1["<<i+1<<" , "<<j+1<<" ] = ";
```


```
        cin>>Matrice1[i][j];
```

```
    }
```

Citirea numarului de linii – n
si de coloane - m ale matricei




Citirea
elemente
lor si
memorar
ea lor in
prima
matrice



7.3. Tablouri bidimensionale (matrici)

```
cout<<"Elementele primei matrici sunt : \n";  
for(i=0; i<n; i++)  
{  
    for(j=0; j<m; j++) cout<<Matrice1[i][j]<<" ";  
    cout<<"\n";  
}
```



Afisarea elementelor
din prima matrice

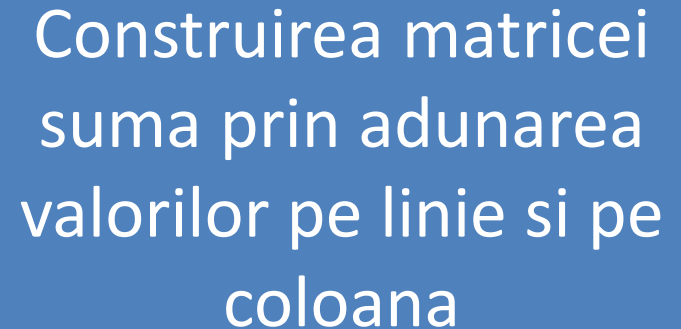
7.3. Tablouri bidimensionale (matrici)

```
for(i=0; i<n; i++){
    for(j=0; j<m; j++){
        cout<<"Matrice2["<<i<<" , "<<j<<" ] = ";
        cin>>Matrice2[i][j];
    }
}

cout<<"Elementele celei de-a doua matrice sunt : \n";
for(i=0; i<n; i++)
{
    for(j=0; j<m; j++) cout<<Matrice2[i][j]<<" ";
    cout<<"\n";
}
```

7.3. Tablouri bidimensionale (matrici)

Construirea matricei
suma prin adunarea
valorilor pe linie si pe
coloana

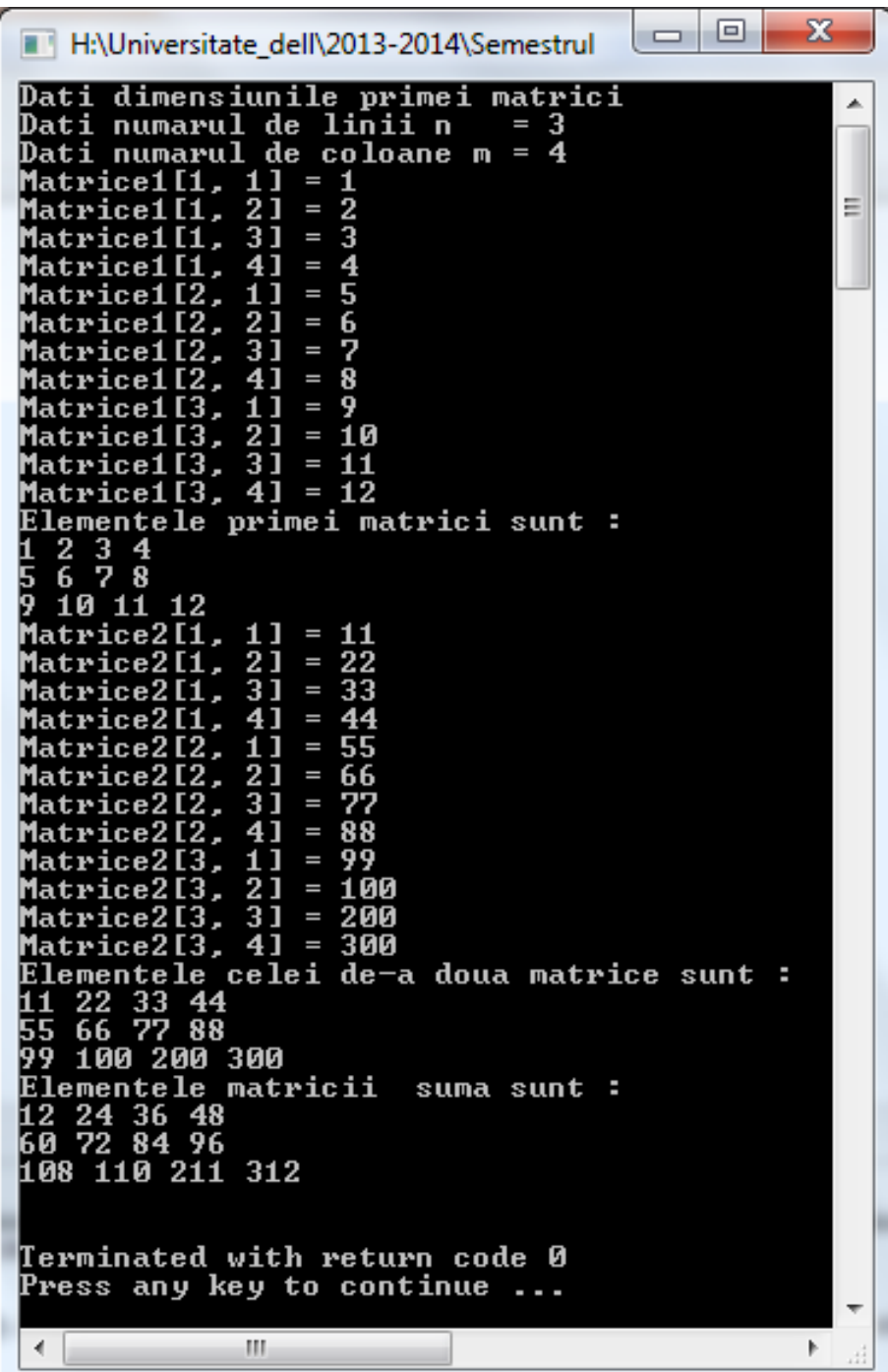


```
for(i=0; i<n; i++)  
    for(j=0; j<m; j++)  
        Matricesuma[i][j]=Matrice1[i][j]+Matrice2[i][j];
```

```
cout<<"Elementele matricii suma sunt : \n";  
    for(i=0; i<n; i++)  
    {  
        for(j=0; j<m; j++) cout<<Matricesuma[i][j]<<" ";  
        cout<<"\n";  
    }  
}
```

7.3. Tablouri bi

Executia
programului pe o
serie de date de
test:



```
H:\Universitate_dell\2013-2014\Semestrul
Dati dimensiunile primei matrici
Dati numarul de linii n = 3
Dati numarul de coloane m = 4
Matrice1[1, 1] = 1
Matrice1[1, 2] = 2
Matrice1[1, 3] = 3
Matrice1[1, 4] = 4
Matrice1[2, 1] = 5
Matrice1[2, 2] = 6
Matrice1[2, 3] = 7
Matrice1[2, 4] = 8
Matrice1[3, 1] = 9
Matrice1[3, 2] = 10
Matrice1[3, 3] = 11
Matrice1[3, 4] = 12
Elementele primei matrici sunt :
1 2 3 4
5 6 7 8
9 10 11 12
Matrice2[1, 1] = 11
Matrice2[1, 2] = 22
Matrice2[1, 3] = 33
Matrice2[1, 4] = 44
Matrice2[2, 1] = 55
Matrice2[2, 2] = 66
Matrice2[2, 3] = 77
Matrice2[2, 4] = 88
Matrice2[3, 1] = 99
Matrice2[3, 2] = 100
Matrice2[3, 3] = 200
Matrice2[3, 4] = 300
Elementele celei de-a doua matrice sunt :
11 22 33 44
55 66 77 88
99 100 200 300
Elementele matricii suma sunt :
12 24 36 48
60 72 84 96
108 110 211 312

Terminated with return code 0
Press any key to continue ...
```

7.3. Tablouri bidimensionale (matrici)

Problema 3:

Se consideră două tablouri bidimensionale (matrici) A și B cu $n \times m$, respectiv $m \times p$ numere întregi.

Se cere să se calculeze **matricea produs**:
 $C = A * B$.

7.3. Tablouri bidimensionale (matrici)

Exemplu:

Date de intrare:

$n=2$, $m=3$ si matricea a:

1 2 3

4 5 6

$m=3$, $p=4$ si matricea b:

1 2 3 4

5 6 7 8

9 10 11 12

Date de iesire:

$n=2$ si $p=3$ si matricea

produs c:

38 44 50 56

83 98 113 128

7.3. Tablouri bidimensionale (matrici)

```
#include<iostream.h>
```

```
int main(void)
```

```
{
```

```
    int a[10][10], b[10][10], c[10][10];
```

```
    int n, m, i, j, k, p;
```

```
    cout<<"Dati dimensiunile matricei A \n";
```

```
    cout<<"Dati numarul de linii n  = ";    cin>>n;
```

```
    cout<<"Dati numarul de coloane m = ";  cin>>m;
```

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

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


```
        {
```

```
            cout<<"a["<<i+1<<", "<<j+1<<" ] = ";
```


```
            cin>>a[i][j];
```

```
        }
```

Citirea numarului de linii – n
si de coloane - m ale matricei

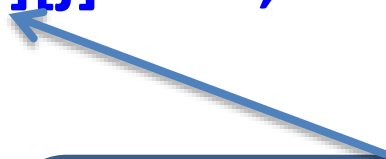


Citirea elementelor
si memorarea lor in
matricea a



7.3. Tablouri bidimensionale (matrici)

```
cout<<"Elementele matricei A sunt : \n";  
for(i=0; i<n; i++)  
{  
    for(j=0; j<m; j++) cout<<a[i][j]<<" ";  
    cout<< "\n";  
}
```

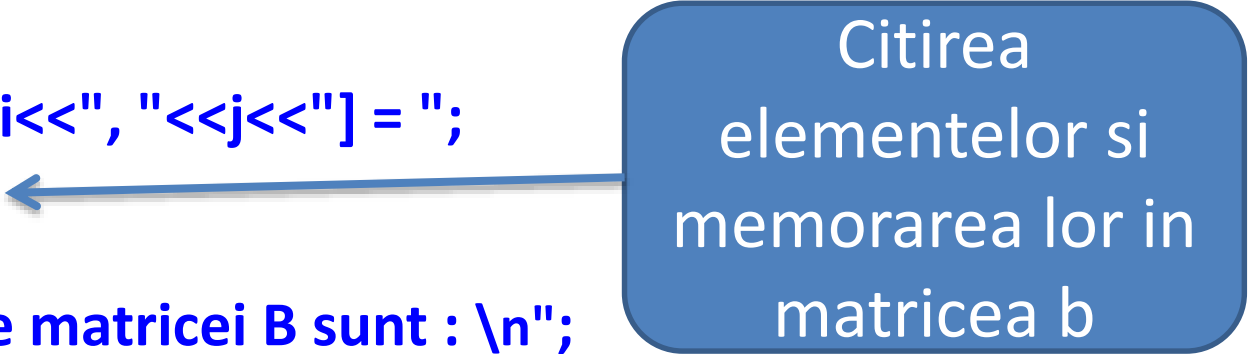


Afisarea elementelor
din matricea a

7.3. Tablouri bidimensionale (matrici)

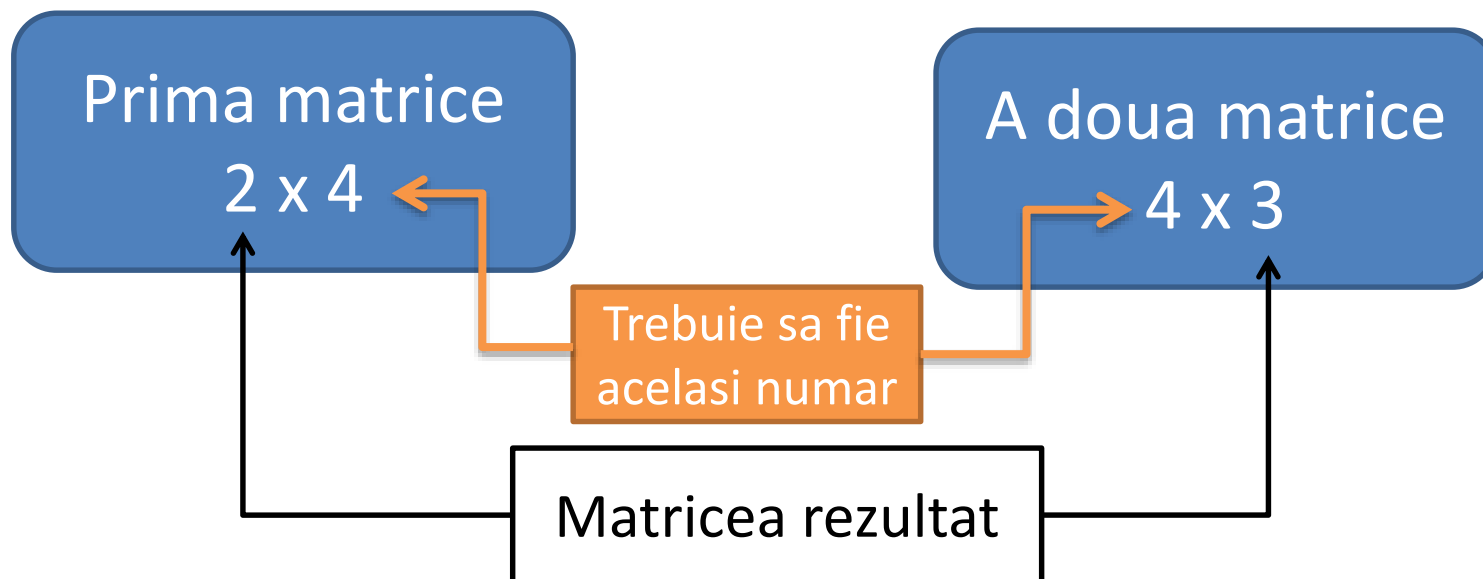
```
cout<<"Dati dimensiunile matricei B \n";
cout<<"Dati numarul de linii m  = ";    cin>>m;
cout<<"Dati numarul de coloane p = ";    cin>>p;
for(i=0; i<m; i++)
    for(j=0; j<p; j++)
    {
        cout<< "b["<<i<<" , "<<j<<" ] = ";
        cin>>b[i][j];
    }
cout<<"Elementele matricei B sunt : \n";
for(i=0; i<m; i++)
{
    for(j=0; j<p; j++) cout<<b[i][j]<<" ";
    cout<<"\n";
}
```

Citirea
elementelor si
memorarea lor in
matricea b



7.3. Tablouri bidimensionale (matrici)

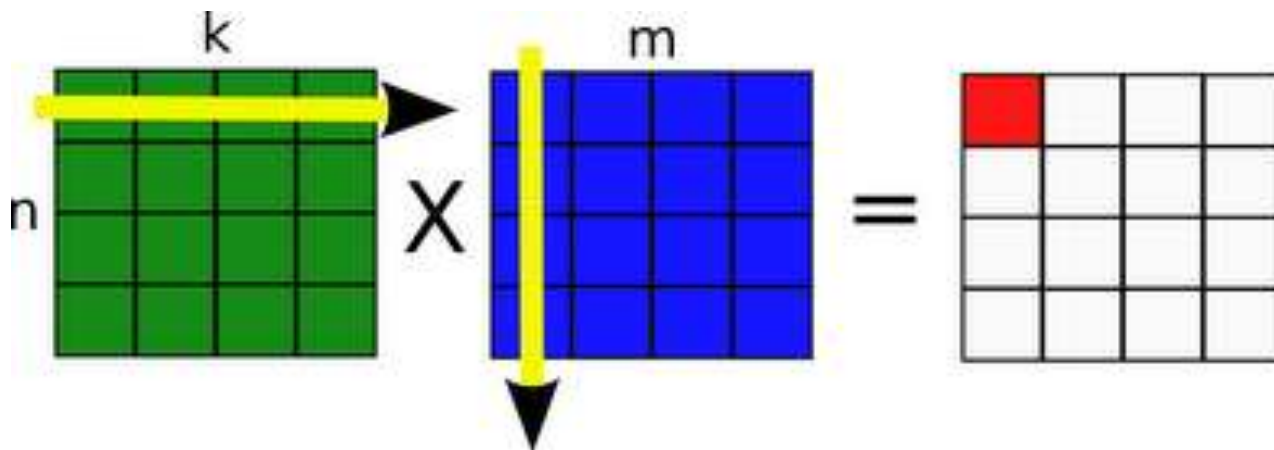
- **Conditie:** *Numarul de coloane din prima matrice trebuie sa fie egal cu numarul de linii din a doua matrice.*



7.3. Tablouri bidimensionale (matrici)

```
for(i=0; i<n; i++)  
  for(j=0; j<p; j++)  
  {  
    c[i][j] = 0;  
    for(k=0; k<m; k++)  
      c[i][j] = c[i][j] + a[i][k] * b[k][j];  
  }
```

Calculul
elementelor
matricei produs c

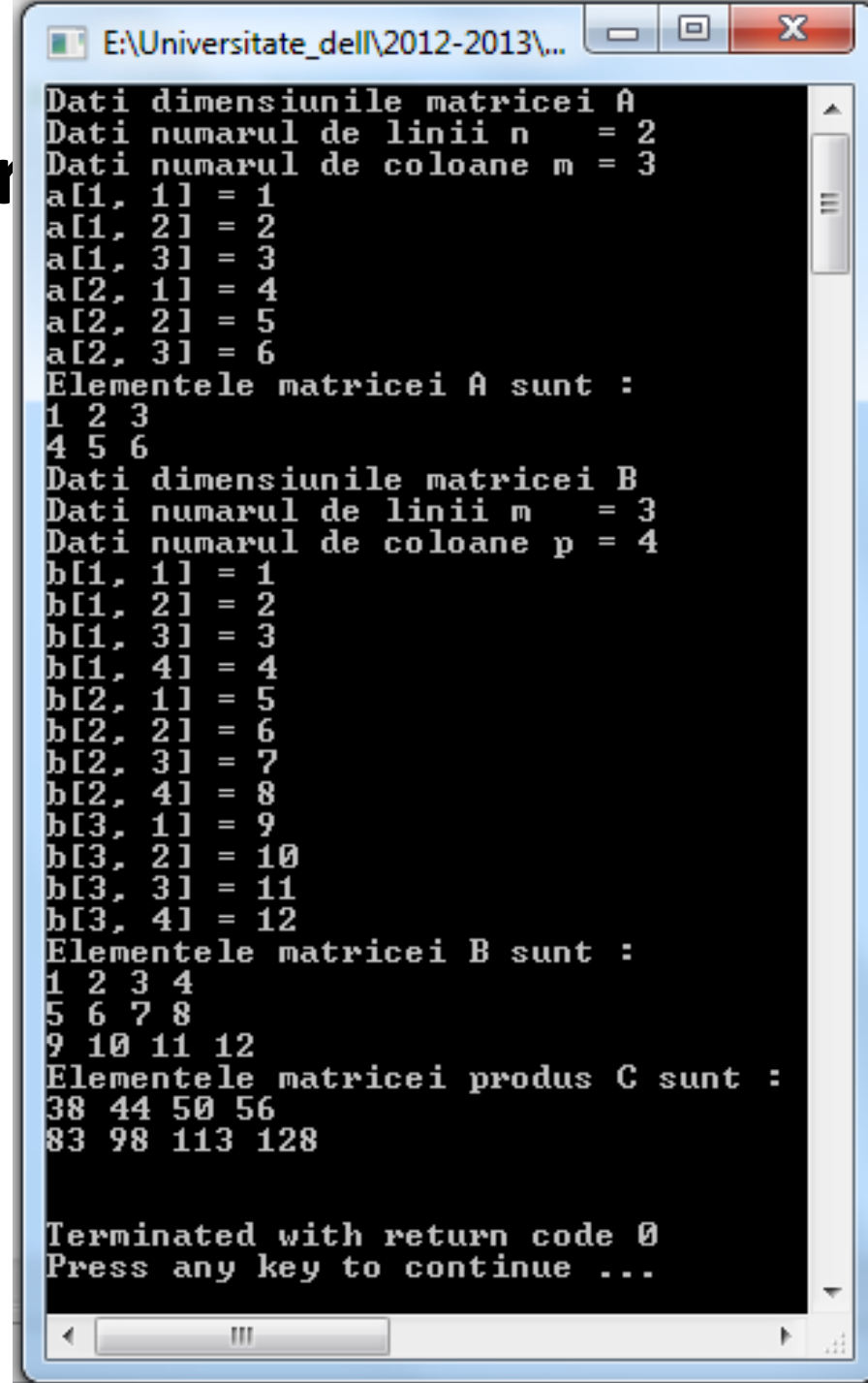


7.3. Tablouri bidimensionale (matrici)

```
cout<<"Elementele matricei produs C sunt : \n";  
for(i=0; i<n; i++) {  
    for(j=0; j<p; j++) cout<<c[i][j]<<" ";  
    cout<<"\n";  
}  
}
```

7.3. Tablouri

Executia
programului pe o
serie de date de
test:



```
E:\Universitate_dell\2012-2013\...
Dati dimensiunile matricei A
Dati numarul de linii n = 2
Dati numarul de coloane m = 3
a[1, 1] = 1
a[1, 2] = 2
a[1, 3] = 3
a[2, 1] = 4
a[2, 2] = 5
a[2, 3] = 6
Elementele matricei A sunt :
1 2 3
4 5 6
Dati dimensiunile matricei B
Dati numarul de linii m = 3
Dati numarul de coloane p = 4
b[1, 1] = 1
b[1, 2] = 2
b[1, 3] = 3
b[1, 4] = 4
b[2, 1] = 5
b[2, 2] = 6
b[2, 3] = 7
b[2, 4] = 8
b[3, 1] = 9
b[3, 2] = 10
b[3, 3] = 11
b[3, 4] = 12
Elementele matricei B sunt :
1 2 3 4
5 6 7 8
9 10 11 12
Elementele matricei produs C sunt :
38 44 50 56
83 98 113 128

Terminated with return code 0
Press any key to continue ...
```

ici)

Probleme rezolvate – tablouri unidimensionale

Enunt:

Se considera un numar natural n . Se cere sa se formeze un vector cu cifrele numarului.

Exemplu:

Date de intrare:

Pentru valoarea: $n = 23416789$

Date de iesire:

Se obtine vectorul cu elementele:

9 8 7 6 1 4 3 2

Probleme rezolvate – tablouri unidimensionale

```
#include<iostream.h>
```

```
int main(void)
```

```
{
```

```
    // declaram variabilele pe care le vom utiliza in program
```

```
    int x[30];
```

```
    long int n, i, m;
```

```
    cout<<"Dati numarul natural n = ";
```

```
    cin>>n;
```

Probleme rezolvate – tablouri unidimensionale

```
i = 0;
while(n != 0){
    x[i] = n % 10;
    i++;
    n = n / 10;
}
m = i-1;
cout<<"\nVectorul cu cifrele numarului este \n";
for(i = 0; i < m; i++){
    cout.width(3);    cout<<x[i];
}
}
```

Probleme rezolvate – tablouri unidimensionale

Executia programului pe o serie de date de test:



```
E:\Universitate_dell\2012-2013\Semestrul  
Dati numarul natural n = 23416789  
Vectorul cu cifrele numarului este  
9 8 7 6 1 4 3 2  
Terminated with return code 0  
Press any key to continue ...
```


Probleme rezolvate – tablouri unidimensionale

Enunt: **Reuniunea a doua multimi**

Sa se scrie un program care sa calculeze reuniunea a doua multimi de cate n , respectiv m numere intregi memorate cu ajutorul vectorilor.

Exemplu:

Date de intrare:

$n = 5$, $a = \{1, 2, 5, 7, 12\}$ si

$m = 4$, $b = \{2, 4, 8, 12\}$

Date de iesire:

se obtine: $k=7$, $c=\{1, 2, 5, 7, 12, 4, 8\}$

Probleme rezolvate – tablouri unidimensionale

```
#include<iostream.h>
```

```
int main(void)
```

```
{
```

```
    int a[100], b[100], c[100], n, m, i, j, k, ok;
```

```
    cout<<"Dati cardinalul multimii A - n = ";
```

```
    cin>>n;
```

```
    cout<<"Dati elementele multimii A \n";
```

```
    for(i = 0; i < n; i++){
```

```
        cout<<"a["<<i+1<<" ] = ";
```

```
        cin>>a[i];
```

```
}
```

Probleme rezolvate – tablouri unidimensionale

```
cout<<"Dati cardinalul multimii B - m = ";
cin>>m;
cout<<"Dati elementele multimii B \n";
for(i = 0; i < m; i++)
{
    cout<<"b["<<i+1<<"] = ";
    cin>>b[i];
}
```

```
// copiem elementele din multimea A in multimea C
for(i = 0; i < n; i++) c[i]=a[i];
k = n;
```

Probleme rezolvate – tablouri unidimensionale

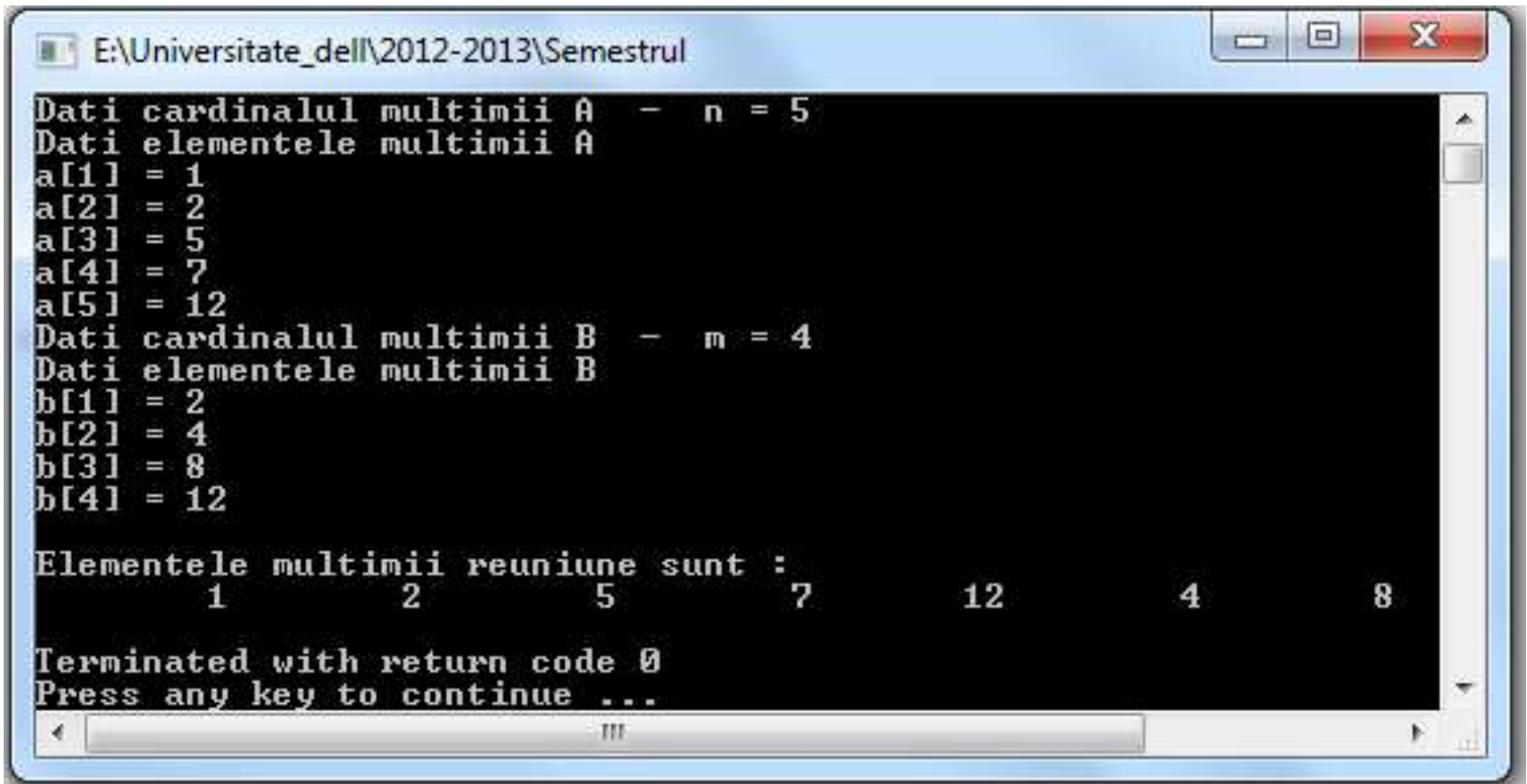
```
for(j = 0; j < m; j++)  
{  
    ok = 1; // variabila care verifica daca un element apartine sau  
un multimii A  
    for(i = 0; i < n ;i++)  
        if(b[j] == a[i]) // daca elementul din multimea B apartine si  
multimii A atunci nu-l adaugam in multimea reuniune - C  
            ok = 0;  
    if(ok == 1)  
    {  
        k++;  
        c[k] = b[j];  
    }  
}
```

Probleme rezolvate – tablouri unidimensionale

```
cout<<"\nElementele multimii reunite sunt : \n";  
    for(i = 0; i < k; i++)  
    {  
        cout.width(9);  
        cout<<c[i];  
    }  
}
```

Probleme rezolvate – tablouri unidimensionale

Executia programului pe o serie de date de test:



```
E:\Universitate_dell\2012-2013\Semestrul
Dati cardinalul multimii A - n = 5
Dati elementele multimii A
a[1] = 1
a[2] = 2
a[3] = 5
a[4] = 7
a[5] = 12
Dati cardinalul multimii B - m = 4
Dati elementele multimii B
b[1] = 2
b[2] = 4
b[3] = 8
b[4] = 12

Elementele multimii reuniune sunt :
      1      2      5      7      12      4      8

Terminated with return code 0
Press any key to continue ...
```

Probleme propuse spre rezolvate – tablouri unidimensionale

1. Intersectia a doua multimi

Sa se scrie un program care sa calculeze intersectia a doua multimi de cate n , respectiv m numere intregi memorate cu ajutorul vectorilor.

Exemplu:

Date de intrare:

$n = 5$ $a = \{1, 2, 5, 7, 12\}$ si

$m = 4$ $b = \{2, 4, 8, 12\}$

Date de iesire:

se obtine: $k = 2$ $d = \{2, 12\}$

Probleme propuse spre rezolvate – tablouri unidimensionale

2. Diferența a două mulțimi

Sa se scrie un program care sa calculeze diferenta a doua multimi de cate n , respectiv m numere intregi memorate cu ajutorul vectorilor.

Exemplu:

Date de intrare:

$n = 5$ $a = \{1, 2, 5, 7, 12\}$ si

$m = 4$ $b = \{2, 4, 8, 12\}$

Date de iesire:

se obtine: $k = 3$ $e = \{1, 5, 7\}$

Probleme propuse spre rezolvate – tablouri unidimensionale

3. Se dă un șir de n numere naturale. Să se afișeze pe două randuri, pe primul rand cele pare și pe al doilea cele impare.

Exemplu:

Date de intrare:

Pentru $n = 10$

si elementele 4 3 2 5 6 8 9 0 1 5

Date de iesire:

4 2 6 8 0

3 5 9 1 5

Probleme propuse spre rezolvate – tablouri unidimensionale

4. Sa se scrie un program care sa introducă n numere într-un vector și să citească un număr d. Să se afișeze acele numere din șirul dat care sunt divizibile cu d.

Exemplu:

Date de intrare:

n=5 si valorile: 5 7 10 23 15, d=5

Date de iesire:

5 10 15

Probleme propuse spre rezolvate – tablouri unidimensionale

5. Se introduc temperaturile măsurate în n zile. Să se scrie un program care să afișeze media temperaturilor negative și media celor pozitive.

Exemplu:

Date de intrare:

$n = 5$ și temperaturile: 23 24 23 25 22

Date de ieșire:

23.40

Referinte bibliografice

Bibliografia necesară cursului:

1. **Adrian Runceanu, Mihaela Runceanu, Noțiuni de programare în limbajul C++,** Academica Brâncuși, Târgu-Jiu, 2012, ISBN 978-973-144-550-2, 483 pagini
2. **Adrian Runceanu, Programarea și utilizarea calculatoarelor,** Editura Academică Brâncuși Targu-Jiu, 2003
3. **Octavian Dogaru, C++ - Teorie și practică, volumul I,** Editura Mirton, Timișoara, 2004
4. O.Catrina, I.Cojocaru, *Turbo C+*, Editura Teora, București, 1993
5. D.Costea, *Inițiere în limbajul C*, Editura Teora, București, 1996
6. K.Jamsa, *C++*, Editura Teora, 1999
7. K.Jamsa & L.Klander, *Totul despre C si C++*, Teora, 2004

Întrebări?