

C++

MATRICE

subiecte pentru Bacalaureat

Ghid pentru rezolvarea în C++
a problemelor cu tablouri
bidimensionale de la
subiectele II și III
propușe la bacalaureat



Teorie



Exemple



Subiecte rezolvate
(2020–2026)

```
01  #include <iostream>
02  #include <vector>
03  using namespace std;
04
05  int main() {
06      int n, m;
07      cin >> n >> m;
08      vector<vector<int>> a(n, vector<int>(m));
09
10      for (int i = 0; i < n; ++i)
11          for (int j = 0; j < m; ++j)
12              cin >> a[i][j];
13
14      // rezolvarea problemei
15
16      return 0;
17  }
```

EDITURA
EVOMIND

Cristina Constantinescu

Matrice - subiecte pentru BACALAUREAT

Constantinescu Cristina

**Ghid pentru rezolvarea în C++ a problemelor cu tablouri
bidimensionale de la subiectele II și III propuse la bacalureat •
*Teorie • Exemple • Subiecte rezolvate (2026 - 2020)***

Editura EVOMIND

ISBN: 978-606-9734-66-7

2026

CUVÂNT ÎNAINTE

Această carte a apărut din dorința de a oferi elevilor de liceu un instrument clar și practic pentru pregătirea exercițiilor privind tablourile bidimensionale pentru proba de Bacalaureat la informatică, în special pentru Subiectul al III-lea – Problema 2- tip obiectiv, dar și pentru exercițiile de la Subiectul II, itemi obiectivi sau semiobiectiv.

Materialul reunește principalele tipuri de probleme pentru prelucrarea tablourilor bidimensionale și exemple de utilizare a acestora în limbajul C++. Fiecare noțiune teoretică este însoțită de cod sursă comentat și de exemple concrete, astfel încât trecerea de la teorie la rezolvarea efectivă a unei probleme să fie cât mai firească și ușor de urmărit.

Ultima parte a cărții reunește un număr mare de subiecte rezolvate, cuprinzând variantele oficiale și testele de antrenament din perioada 2026–2020. Parcurgerea lor, în ordine sau selectiv, în funcție de nevoile fiecărui elev, oferă exercițiul necesar pentru a aborda cu încredere proba din ziua examenului.

Recomand ca această carte să fie folosită activ. Studiați codul problemelor cu atenție, testați-l pe calculator și încercați și alte implementări ale problemelor.

Mult succes la Bacalaureat!

Cristina Constantinescu

Contents

1. NOȚIUNI TEORETICE — MATRICE ÎN C++	6
1.1 Declararea, citirea și afișarea unei matrice	6
1.2 Parcurgerea pe linii și pe coloane	7
1.3 Diagonalele unei matrice pătratice	7
1.4 Elemente simetrice și matrice simetrică	9
1.5 Suma / numărul elementelor cu o proprietate dată	9
1.6 Determinarea elementului minim/maxim și a poziției sale	10
1.7 Interschimbarea a două linii / a două coloane	10
1.8 Transpusa unei matrice	12
1.9 Eliminarea unei linii/coloane și rearanjarea matricei	12
1.10 Inserarea unei noi linii/ coloane într-o matrice	13
1.11 Rotirea unei matrice pătratice cu 90°	14
1.12 Prelucrarea conturului (marginea) unei matrice	14
1.13 Bordarea unei matrice	15
1.1 Deplasare dintr-un punct dat la vecinii acestuia	16
1. PROBLEME DATE LA BACALAUREAT	18
2.1 Subiect 2026. Varianta 3. Sub II. 5	18
2.2 Subiect 2026. Varianta 5	19
2.3 Subiect 2026. Varianta 4	20
2.4 Subiect 2026. SIMULARE. Sub I. 2	21
2.5 Subiect 2026. Varianta MODEL	22
2.6 Subiect 2025. Varianta 4	23
2.7 Subiect 2025. Varianta 7	24
2.8 Subiect 2025. Varianta 4. Sub I. 3	25
2.9 Subiect 2025. Varianta 6. Sub I. 4	25
2.10 Subiect 2025. SIMULARE	25
2.11 Subiect 2025. Varianta MODEL	26
2.12 Subiect 2024. Varianta 8	28
2.13 Subiect 2024. Varianta 8. Sub II.3	29
2.14 Subiect 2024. Varianta 4	30
2.15 Subiect 2024. Varianta SIMULARE. Sub. II.3	31
2.16 Subiect 2023. Varianta 7	32

2.17	Subiect 2023. Varianta 5.....	33
2.18	Subiect 2023. Varianta 6.....	34
2.19	Subiect 2023. Varianta SIMULARE. Sub II.3.....	35
2.20	Subiect 2023. Varianta MODEL.....	36
2.21	Subiect 2022. Varianta 1.....	37
2.22	Subiect 2022. Varianta 4.....	38
2.23	Subiect 2022. Varianta SIMULARE. Sub. II.3.....	39
2.24	Subiect 2021. Varianta 4.....	40
2.25	Subiect 2021. Varianta 7.....	41
2.26	Subiect 2021. Varianta SIMULARE.....	42
2.27	Subiect 2021. Varianta TEST ANTRENAMENT 1.....	43
2.28	Subiect 2021. Varianta TEST ANTRENAMENT 2.....	44
2.29	Subiect 2021. Varianta TEST ANTRENAMENT 3.....	45
2.30	Subiect 2021. Varianta TEST ANTRENAMENT 4.....	46
2.31	Subiect 2021. Varianta TEST ANTRENAMENT 5.....	47
2.32	Subiect 2021. Varianta TEST ANTRENAMENT 6.....	48
2.33	Subiect 2021. Varianta TEST ANTRENAMENT 7.....	49
2.34	Subiect 2021. Varianta TEST ANTRENAMENT 8.....	50
2.35	Subiect 2021. Varianta TEST ANTRENAMENT 9.....	50
2.36	Subiect 2021. Varianta TEST ANTRENAMENT 10.....	51
2.37	Subiect 2021. Varianta TEST ANTRENAMENT 11.....	52
2.38	Subiect 2021. Varianta TEST ANTRENAMENT 12.....	53
2.39	Subiect 2020. Varianta 5. Sub. II. 3.....	54
2.40	Subiect 2020. Varianta 2.....	54
2.41	Subiect 2020. Varianta 6.....	55
2.42	Subiect 2020. Varianta TEST ANTRENAMENT 1.....	56
2.43	Subiect 2020. Varianta TEST ANTRENAMENT 2. Sub II.3.....	57
2.44	Subiect 2020. Varianta TEST ANTRENAMENT 3.....	57
2.45	Subiect 2020. Varianta TEST ANTRENAMENT 4.....	58
2.46	Subiect 2020. Varianta TEST ANTRENAMENT 5. Sub. II.3.....	59
2.47	Subiect 2020. Varianta TEST ANTRENAMENT 6. Sub. II.3.....	59
2.48	Subiect 2020. Varianta TEST ANTRENAMENT 7.....	60
2.49	Subiect 2020. Varianta TEST ANTRENAMENT 8.....	61

2.50	Subiect 2020. Varianta TEST ANTRENAMENT 10. Sub II.3.....	62
2.51	Subiect 2020. Varianta TEST ANTRENAMENT 11.....	62
2.52	Subiect 2020. Varianta TEST ANTRENAMENT 13.....	63
2.53	Subiect 2020. Varianta TEST ANTRENAMENT 14.....	64
2.54	Subiect 2020. Varianta TEST ANTRENAMENT 15.....	64
2.55	Subiect 2020. Varianta TEST ANTRENAMENT 16.....	65
2.56	Subiect 2020. Varianta TEST ANTRENAMENT 17.....	66
2.57	Subiect 2020. Varianta TEST ANTRENAMENT 19.....	66
2.58	Subiect 2020. Varianta TEST ANTRENAMENT 20.....	67

1.

1. NOȚIUNI TEORETICE — MATRICE ÎN C++

1.1 Declararea, citirea și afișarea unei matrice

O matrice (tablou bidimensional) cu m linii și n coloane se declară static ca un tablou de tablouri. Indicii de linie și de coloană pornesc, în C++, de la 0.

```
int a[100][100];
int m, n;
cin >> m >> n;
for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
        cin >> a[i][j];
```

Afișarea se face linie cu linie, cu trecere la linie nouă (endl) după fiecare linie a matricei:

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

În practică, matricea poate fi parcursă cu și cu indicii de la 1. De cele mai multe ori, aceasta este utilă, dacă pe linia 0 sau pe coloana 0 trebuie să se păstreze alte elemente, de exemplu cea mai mică valoare de pe fiecare linie etc.

```
int a[100][100];
int m, n;
cin >> m >> n;
for (int i = 1; i <= m; i++)
    for (int j = 1; j <= n; j++)
        cin >> a[i][j];
```

Afișarea, și în această situație se face linie cu linie, cu trecere la linie nouă (endl) după fiecare linie a matricei:

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

1.2 Parcurgerea pe linii și pe coloane

Parcurgerea "pe linii" se realizează ca la capitolul anterior, cu i indice pentru linie și j indice pentru coloană și vizitează elementele în ordinea $a[0][0]$, $a[0][1]$, ..., $a[0][n-1]$, $a[1][0]$...

Parcurgerea "pe coloane" se obține dacă se folosește mai întâi indicele j și apoi indicele i :

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

1.3 Diagonalele unei matrice pătratică

Pentru o matrice pătratică (adică numărul de linii egal cu numărul de coloane, ambele având valoarea n): elementele diagonalei principale sunt $a[i][i]$, $i=0..n-1$, iar elementele diagonalei secundare sunt $a[i][n-1-i]$, $i=0..n-1$.

Este posibil să avem un element pe ambele diagonale doar dacă n este impar. Acest element verifică faptul că $i = n-1-i$, adică $i = (n-1)/2$.

Pentru a calcula suma elementelor pe cele două diagonale, este suficient să parcurgem liniile matricei și să accesăm doar elementele de pe diagonale.

```
int sp = 0, ss = 0;
for (int i = 0; i < n; i++)
{
    sp = sp + a[i][i];      // suma de pe diagonala principala
    ss = ss - a[i][n-1-i]; // suma de pe diagonala secundara
}
```

Pentru o matrice pătratică $n \times n$, se pun în evidență:

- **elementele diagonalei principale:** $i = j$
- **elementele diagonalei secundare:** $j = n - 1 - i$
- **elementele aflate deasupra ambelor diagonale:** $i \leq j \ \&\& \ i + j \leq n - 1$
- **elementele aflate sub ambele diagonale:** $i \geq j \ \&\& \ i + j \geq n - 1$
- **elementele aflate între diagonale (inclusiv diagonalele):**
 - o pentru partea stângă: $i \geq j \ \&\& \ i + j \leq n - 1$
 - o pentru partea dreaptă: $i \leq j \ \&\& \ i + j \geq n - 1$

Pentru a calcula suma elementelor dintre cele două diagonale și de pe diagonale, este necesară parcurgerea matricei și verificarea în care zonă se află elementul curent.

```
int sp = 0, ss = 0, sus = 0, jos = 0, st = 0, dr = 0;
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n; j++)
    {
        if (i == j)
            sp = sp + a[i][j];
        if (i + j == n - 1)
            ss = ss + a[i][j];
        if (i < j && i + j < n - 1)
            sus = sus + a[i][j];
        if (i > j && i + j > n - 1)
            jos = jos + a[i][j];
        if (i > j && i + j < n - 1)
            st = st + a[i][j];
        if (i < j && i + j > n - 1)
            dr = dr + a[i][j];
    }
}
```

```
cout << "Diagonala principala = " << sp << endl;
cout << "Diagonala secundara = " << ss << endl;
cout << "Sus = " << sus << endl;
cout << "Jos = " << jos << endl;
cout << "Stanga = " << st << endl;
cout << "Dreapta = " << dr << endl;
```

Acesta este algoritmul standard pentru împărțirea unei matrice pătratice în 6 regiuni: cele două diagonale și cele patru zone dintre ele.

1.4 Elemente simetrice și matrice simetrică

Elementul simetric față de diagonala principală al lui $a[i][j]$ este $a[j][i]$. O matrice pătratică este simetrică dacă $a[i][j] = a[j][i]$ pentru orice i și j . Este suficientă parcurgerea matricei deasupra diagonalei principale, adică i să ia valori de la 0 la $n-1$, iar pentru fiecare i , j să ia valori de la 0 la $i-1$ și să se verifice dacă $a[i][j]$ este egal cu $a[j][i]$.

```
bool ok = true;
for (int i = 0; i < n-1 && ok; i++)
    for (int j = 0; j < i && ok; j++)
        if (a[i][j] != a[j][i])
            ok = false;
if (ok)
    cout << "Matricea este simetrica fata de diagonala principala";
else
    cout << "Matricea nu este simetrica fata de diagonala principala";
```

Elementul simetric față de diagonala secundară al lui $a[i][j]$ este $a[n-1-j][n-1-i]$. Similar se testează dacă o matrice pătratică este simetrică față de diagonala secundară.

```
bool ok = true;
for (int i = 0; i < n && ok; i++)
    for (int j = 0; j < n && ok; j++)
        if (i + j < n - 1) // doar o parte a matricei
            if (a[i][j] != a[n - 1 - j][n - 1 - i])
                ok = false;
if (ok)
    cout << "Matricea este simetrica fata de diagonala secundara";
else
    cout << "Matricea nu este simetrica fata de diagonala secundara";
```

1.5 Suma / numărul elementelor cu o proprietate dată

Modelul general presupune parcurgerea completă a matricei cu m linii și n coloane și testarea unei condiții pentru fiecare element, cu acumularea rezultatului într-o variabilă (sumă, produs sau contor). Modelul se poate aplica pe întreaga matrice, pe o singură linie (i fixat) sau pe o singură coloană (j fixat):

```
int s = 0, nr = 0;
for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
        if (a[i][j] % 2 == 0) // conditia dorita
        {
            s = s + a[i][j];
            nr++;
        }
```


1.6 Determinarea elementului minim/maxim și a poziției sale

Se inițializează maximul (minimul) cu primul element și se comparăm succesiv cu restul, reținând și poziția (linia și coloana) elementului găsit:

```
int maxim = a[0][0], linie = -1, coloana = -1;
for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
        if (a[i][j] > maxim)
        {
            maxim = a[i][j];
            linie = i;
            coloana = j;
        }
```

Pentru determinarea minimului/maximului pe o linie i dată, se parcurge doar j = 0..n-1 cu i fixat; pentru o coloană j dată, se parcurge doar i = 0..m-1 cu j fixat.

1.7 Interschimbarea a două linii / a două coloane

Se realizează interschimbând element cu element fiecare element de pe două linii date L1 și L2, respectiv două coloane date C1 și C2, dar și direct:

```
// interschimbarea liniilor L1 si L2
for (int j = 0; j < n; j++)
    swap(a[L1][j], a[L2][j]);

// interschimbarea coloanelor C1 si C2
for (int i = 0; i < m; i++)
    swap(a[i][C1], a[i][C2]);
```

```

#include <iostream>
using namespace std;
int main()
{
    int a[100][100];
    int m, n;
    cin >> m >> n;
    /// Citirea matricei
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            cin >> a[i][j];
    /// Citirea liniilor care se interschimba
    int L1, L2;
    cin >> L1 >> L2;
    /// Interschimbarea liniilor
    for (int j = 0; j < n; j++)
        swap(a[L1][j], a[L2][j]);
    /// Afisarea matricei
    for (int i = 0; i < m; i++)
    {
        for (int j = 0; j < n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}

```

"C:\Users\User1\Desktop

```

3 4
1 2 3 4
5 6 7 8
9 10 11 12
0 2
9 10 11 12
5 6 7 8
1 2 3 4

Process returned 0
Press any key to c

```

1.8 Transpusa unei matrice

Transpusa unei matrice cu m linii și n coloane este o matrice cu n linii și m coloane, în care $t[j][i] = a[i][j]$.

Dacă matricea nu este pătratică, transpusa se construiește obligatoriu într-un tablou separat:

```
int t[100][100];
for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
        t[j][i] = a[i][j];
// t are n linii si m coloane
```

Pentru o matrice pătratică, transpunerea se poate face "pe loc", interschimbând doar elementele situate deasupra diagonalei principale cu cele simetrice lor:

```
for (int i = 0; i < n; i++)
    for (int j = i+1; j < n; j++)
        swap(a[i][j], a[j][i]);
```

1.9 Eliminarea unei linii/coloane și rearanjarea matricei

Eliminarea elementelor unei linii/coloane (sau ale unei diagonale) se poate realiza prin păstrarea elementelor care nu se șterg. Elementele păstrate se copiază, în ordinea parcurgerii, pe linii, într-un tablou auxiliar unidimensional, apoi acesta este redistribuit în matricea rezultat cu noul număr de linii/ coloane.

Pentru ștergerea elementelor de pe diagonala secundară a matricei, se poate construi un tablou unidimensional (v) în care se pun toate elementele, apoi se construiește un tablou bidimensional cu n linii și n-1 coloane în care se aduc elementele din vector.

```
int v[10000], k = 0;
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
        if (j != n-1-i) // se elimina, de exemplu, diag. secundara
            v[k++] = a[i][j];
// redistribuire in matricea b, cu n-1 elemente valide pe (majoritatea) liniilor
int p = 0;
for (int i = 0; i < n && p < k; i++)
    for (int j = 0; j < n-1 && p < k; j++)
        b[i][j] = v[p++];
```

O altă abordare, permite deplasarea elementelor aflate la dreapta diagonalei secundare, în aceeași matrice, iar afișarea matricei se face până la coloana n-2, ultima și penultima coloană având aceleași valori.

```
// eliminarea diagonalei secundare prin deplasare la stânga
for (int i = 0; i < n; i++)
    for (int j = n - 1 - i; j < n - 1; j++)
        a[i][j] = a[i][j + 1];
// afișare (se afișează doar n-1 coloane)
for (int i = 0; i < n; i++)
{
    for (int j = 0; j < n - 1; j++)
        cout << a[i][j] << " ";
    cout << '\n';
}
```

Pentru ștergerea unei linii sau coloane, se poate proceda similar. Deplasez în sus toate liniile următoare celei ce trebuie să fie ștersă sau deplasez spre stânga toate coloanele ce urmează celei ce trebuie să fie ștersă.

1.10 Inserarea unei noi linii/ coloane într-o matrice

Se consideră $v_1, v_2, \dots, v_n$. Se dorește inserarea elementelor vectorului v pe o anumită linie (k) data a matricei. Pentru aceasta elementele liniei $m-1$ se deplasează pe linia următoare, elementele liniei $m-2$ pe linia $m-1$, ș.a.m.d. elementele liniei k pe linia $k+1$ în matricea a cu m linii și n coloane.

```
// k = poziția pe care se inserează ( $0 \leq k \leq m$ )
for (int i = m; i > k; i--)
    for (int j = 0; j < n; j++)
        a[i][j] = a[i - 1][j];
// copierea vectorului pe linia k
for (int j = 0; j < n; j++)
    a[k][j] = v[j+1];
m++; // matricea are acum o linie în plus
La fel se procedează și pentru inserarea elementelor pe o coloană data.
Adunarea și înmulțirea a două matrice
Suma a două matrice cu aceleași dimensiuni  $m \times n$  se calculează element cu element:
c[i][j] = a[i][j] + b[i][j].
Produsul a două matrice, a are dimensiunea  $m \times p$ , b de dimensiune  $p \times n$ , se calculează astfel:
elementul c[i][j] este suma produselor elementelor de pe linia i a lui a cu cele de pe coloana j a lui b:
for (int i = 0; i < m; i++)
    for (int j = 0; j < n; j++)
    {
        c[i][j] = 0;
        for (int k = 0; k < p; k++)
            c[i][j] = c[i][j] + a[i][k] * b[k][j];
    }
```

1.11 Rotirea unei matrice pătrate cu 90°

Rotirea la dreapta cu 90° a unei matrice $n \times n$ plasează elementul $a[i][j]$ pe poziția $[j][n-1-i]$ în matricea rezultat:

```
int r[100][100];
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
        r[j][n-1-i] = a[i][j];
```

1.12 Prelucrarea conturului (marginea) unei matrice

Conturul unei matrice cu m linii și n coloane este format din elementele primei și ultimei linii (integral) și din elementele primei și ultimei coloane, fără colțurile deja folosite.

Problema de mai jos calculează suma elementelor de pe conturul matricei.

```
int s = 0;
// parcurg linia 0 si linia m-1
for (int j = 0; j < n; j++)
    s = s + a[0][j] + a[m-1][j];

// parcurg coloana 0 și coloana n-1
for (int i = 1; i < m-1; i++)
    s = s + a[i][0] + a[i][n-1];
```

1.13 Bordarea unei matrice

În multe probleme, este indicat ca elementele utile să fie păstrate începând de pe linia 1 și coloana 1, astfel încât linia și coloana 0 să fie rezervate altor valori.

În situația în care se cer identificarea elementelor din matrice care respectă o proprietate în raport cu elementele vecine acestora, de exemplu ecele elemente care sunt mai mici decât toți vecinii acestora, pentru un cod mai lizibil, este recomandat să se “bordeze” matricea cu cea mai mare valoare valoare maximă.

```
#include <iostream>
#include <climits>
using namespace std;
int main()
{
    int n, m;
    cin >> n >> m;
    int a[102][102];
    // Bordarea matricei cu valoarea maximă
    /// Linia 0 si linia n+1
    for (int j = 0; j <= m + 1; j++)
    {
        a[0][j] = INT_MAX;
        a[n + 1][j] = INT_MAX;
    }
    /// Coloana 0 si coloana m+1
    for (int i = 1; i <= n; i++)
    {
        a[i][0] = INT_MAX;
        a[i][m + 1] = INT_MAX;
    }
    /// Citirea elementelor începând de la (1,1)
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= m; j++)
            cin >> a[i][j];
    /// Afisarea elementelor mai mici decât toți cei 4 vecini
    cout << "Elementele cautate sunt:\n";
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= m; j++)
            if (a[i][j] < a[i - 1][j] && // sus
                a[i][j] < a[i + 1][j] && // jos
                a[i][j] < a[i][j - 1] && // stanga
                a[i][j] < a[i][j + 1]) // dreapta
                cout << a[i][j] << " ";
    return 0;
}
```

"C:\Users\User1\Desktop\Dan Barbilian\matrice pct sa\bin\Debug\matrice pct sa.exe"

```
3 3
5 3 6
7 1 8
4 9 2
Elementele cautate sunt:
1 4 2
```

1.1 Deplasare dintr-un punct dat la vecinii acestuia.

Într-o matrice , poziția unui element este definită de coordonatele (linie, coloană) sau (r, c).

Se poate folosi abordarea de la problema anterioară, dar, de regulă, se folosesc doi vectori (sau tablouri) care să asigure trecerea la o celulă vecină, adăugăm niște valori "deplasare" la coordonatele curente.

- **dr** modific pe linie (rând)
- **dc** modific pe coloană

Direcții de deplasare

1. Deplasare în 4 direcții (Nord, Sud, Est, Vest)

Pentru a accesa vecinii de sus, jos, stânga, dreapta, voi declara:

int **dr**[] = {-1, 1, 0, 0}; // Sus, Jos, Stânga, Dreapta

int **dc**[] = {0, 0, -1, 1}; // Sus, Jos, Stânga, Dreapta

- $r+dr[0], c+dc[0] \Rightarrow$ Sus ($r-1, c + 0$) $\Rightarrow a[r+dr[0]] [c+dc[0]]$ este punctul aflat pe aceeași coloană și linia de deasupra.
- $r+dr[1], c+dc[1] \Rightarrow$ Jos ($r+1, c + 0$) $\Rightarrow a[r+dr[1]] [c+dc[1]]$ este punctul aflat pe aceeași coloană și linia de jos.
- $r+dr[2], c+dc[2] \Rightarrow$ Stânga ($(r+0, c -1)$) $\Rightarrow a[r+dr[2]] [c+dc[2]]$ este punctul aflat pe aceeași linie și o coloană la stânga.
- $r+dr[3], c+dc[3] \Rightarrow$ Dreapta ($((r+0, c +1)$) $\Rightarrow a[r+dr[3]] [c+dc[3]]$ este punctul aflat pe aceeași linie și coloana de la dreapta.

Problema rezolvată la capitolul anterior poate să aibă și următoarea soluție:

```
#include <iostream>
using namespace std;
int a[102][102];
int dr[] = {-1, 1, 0, 0}; // pentru deplasare sus, jos, stânga, dreapta
int dc[] = {0, 0, -1, 1};
int main() {
    int n, m;
    cin >> n >> m;
    // Citirea elementelor începând de la (1,1)
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= m; j++)
            cin >> a[i][j];
    cout << "Elementele mai mari decat toti vecinii sunt:\n";
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= m; j++) {
            bool maxim = 1;
            // Verificăm cei 4 vecini
            for (int k = 0; k < 4; k++) {
                int lin = i + dr[k];
                int col = j + dc[k];
                if (a[i][j] <= a[lin][col]) {
                    maxim = 0;
                    break;
                }
            }
            if (maxim)
                cout << a[i][j] << " ";
        }
    return 0;
}
```

"C:\Users\User1\Desktop\Dan Barbilian\matrice pct sa\bin\Debug\matrice pct sa.exe"

```
3 3
5 3 6
7 1 8
4 9 2
Elementele cautate sunt:
1 4 2
```

Pentru Deplasare în 8 direcții se po folosi vectorii de deplasare:

```
int dr[] = {-1, -1, -1, 0, 0, 1, 1, 1};
int dc[] = {-1, 0, 1, -1, 1, -1, 0, 1};
```


1. PROBLEME DATE LA BACALAUREAT

2.1 Subiect 2026. Varianta 3. Sub II. 5

Variabilele `i` și `j` sunt de tip întreg, iar variabila `a` memorează un tablou bidimensional cu 5 linii și 5 coloane, numerotate de la 0 la 4, având inițial toate elementele nule.

Fără a utiliza alte variabile, scrieți secvența C/C++ de mai jos, înlocuind punctele de suspensie cu instrucțiuni adecvate, dintre care cel mult patru atribuiri, astfel încât, în urma executării secvenței obținute, tabloul memorat în variabila `a` să aibă elementele alăturate.

```
for (i=0; i<5; i++)  
    for (j=0; j<5; j++)
```

```
        .....
```

(6p.)

```
for (int i=0; i<5; i++)  
    for (int j=0; j<5; j++)  
        if (j%2==0)  
            a[i][j]=5*j+i;  
        else  
            a[i][j]=5*j+4-i;
```

0	9	10	19	20
1	8	11	18	21
2	7	12	17	22
3	6	13	16	23
4	5	14	15	24

2.2 Subiect 2026. Varianta 5

Într-un tablou bidimensional, două coloane sunt numite **stabile** dacă suma valorilor de pe una dintre ele este egală cu suma valorilor de pe cealaltă, iar cele două coloane sunt consecutive în tablou.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[1, 20]$, m și n , și $m \cdot n$ numere naturale din intervalul $[0, 10]$, elemente ale unui tablou bidimensional cu m linii și n coloane, numerotate începând de la 0. Programul afișează pe ecran toate perechile de numere x și y ($x < y$), pentru care coloanele x și y sunt stabile în tabloul dat, sau mesajul **nu exista**, dacă nu există astfel de coloane.

Fiecare pereche se afișează pe câte o linie a ecranului, cu cele două valori în ordine strict crescătoare, separate prin câte un spațiu.

Exemplu: pentru $m=6$, $n=7$ și tabloul alăturat, se afișează pe ecran perechile de mai jos, nu neapărat în această ordine

```
0 1
1 2
5 6
```

2	6	1	0	4	0	0
0	3	5	2	3	1	0
2	0	0	0	1	0	1
6	0	5	0	6	0	1
1	3	0	0	0	1	0
1	0	1	0	2	0	0

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, a[20][20];
    cin >> m >> n;
    for(int i = 0; i < m; i++)
        for(int j = 0; j < n; j++)
            cin >> a[i][j];
    bool ok = false;
    for(int j = 0; j < n - 1; j++)
    {
        int s1 = 0, s2 = 0;

        for(int i = 0; i < m; i++)
        {
            s1 += a[i][j];
            s2 += a[i][j + 1];
        }
        if(s1 == s2)
        {
            cout << j << " " << j + 1 << endl;
            ok = true;
        }
    }
    if(ok == 0)
        cout << "nu exista";
    return 0;
}
```

2.3 Subiect 2026. Varianta 4

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 50]$, m și n , și elementele unui tablou bidimensional cu m linii și n coloane, numere naturale din intervalul $[0, 10^2]$. Programul transformă tabloul în memorie, interschimbând, pentru fiecare coloană, elementul său de valoare minimă cu primul element al coloanei, dacă acest minim este par, sau cu ultimul element al coloanei, dacă acest minim este impar. Dacă pe o coloană sunt mai multe elemente de valoare minimă, se alege doar unul dintre acestea. Tabloul obținut este afișat pe ecran, linie cu linie, de sus în jos, fiecare linie a tabloului pe câte o linie a ecranului, cu elementele parcurse de la stânga la dreapta, separate prin câte un spațiu.

Exemplu: pentru $m=4$, $n=5$ și

7	5	3	0	9
1	2	1	9	7
2	2	0	1	5
5	6	1	2	8

se poate obține
tabloul

7	2	0	0	9
5	5	1	9	7
2	2	3	1	8
1	6	1	2	5

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, a[50][50];
    cin >> m >> n;
    for(int i = 0; i < m; i++)
        for(int j = 0; j < n; j++)
            cin >> a[i][j];
    for(int j = 0; j < n; j++)
    {
        int minim = a[0][j], poz = 0;
        for(int i = 1; i < m; i++)
            if(a[i][j] < minim)
            {
                minim = a[i][j];
                poz = i;
            }
        if(minim % 2 == 0)
        {
            int aux = a[0][j];
            a[0][j] = a[poz][j];
            a[poz][j] = aux;
        }
        else
        {
            int aux = a[m-1][j];
            a[m-1][j] = a[poz][j];
            a[poz][j] = aux;
        }
    }
    for(int i = 0; i < m; i++)
    {
        for(int j = 0; j < n; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
    return 0;
}
```

2.4 Subiect 2026. SIMULARE. Sub I. 2

2. Un mozaic este alcătuit din plăcuțe de formă pătrată, cu dimensiuni egale, de culoare albă sau roșie, așezate pe două rânduri și aliniate ca în exemplu. Mozaicul are un model în zig-zag dacă oricare două plăcuțe cu o latură comună au culoare diferite. Se știe că plăcuțele de pe latura din dreapta au culori diferite.

Variabilele j și ok sunt întregi, iar variabila a este un tablou bidimensional cu două linii și 12 coloane, cu elemente de tip `char`, în care fiecare linie, în ordine, memorează succesiunea de plăcuțe aflate pe un rând al mozaicului, notându-se cu litera **R** cele roșii și cu litera **A** cele albe. Liniile și coloanele sunt numerotate începând de la 0. Indicați expresia care poate înlocui punctele de suspensie astfel încât, în urma executării secvenței C/C++ obținute, variabila ok să aibă valoarea 1, dacă mozaicul are un model în zig-zag, sau valoarea 0 în caz contrar.

Exemplu: în urma executării secvenței pentru tabloul de mai jos, unde sunt evidențiate două zone în care nu se respectă modelul, $ok=0$.

```
ok=1;
for(j=0; j<11; j++)
    if(.....) ok=0;
```

R	A	A	R	A	R	A	R	A	R	A	R
A	R	R	A	R	A	R	A	R	R	R	A

- a. $a[0][j]==a[0][j-1] \ || \ a[0][j]==a[1][j-1]$ b. $a[0][j]==a[0][j+1] \ || \ a[0][j]==a[1][j+1]$
 c. $a[0][j]==a[0][j-1] \ || \ a[0][j]==a[1][j+1]$ d. $a[0][j]==a[0][j+1] \ || \ a[0][j]==a[1][j]$

Într-un model **zig-zag**, două plăcuțe care au o latură comună trebuie să aibă culori diferite.

Răspunsul corect este d.

2.5 Subiect 2026. Varianta MODEL

Terenul alocat unui târg de jucării artisanale a fost împărțit în zone dispuse ca elementele unui tablou bidimensional cu număr egal de linii și coloane. Pentru a putea fi testate de copii, fiecare jucărie are alocată o zonă și este identificată printr-un număr natural. Organizatorii au hotărât eliminarea tuturor jucăriilor plasate pe diagonala secundară a tabloului și rearanjarea celor păstrate, prin deplasarea unora dintre acestea cu câte o poziție spre stânga, astfel încât să nu existe zone intermediare libere, ca în exemplu.

Scrieți un program C/C++ care citește de la tastatură numere naturale din intervalul $[2, 10^2]$, și anume n și elementele unui tablou bidimensional cu n linii și n coloane, reprezentând numerele de identificare ale jucăriilor, în ordinea plasării lor inițiale pe teren. Programul modifică apoi tabloul în memorie corespunzător hotărârii organizatorilor și afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.

Exemplu: pentru $n=4$ și tabloul

20	25	20	26
36	16	48	52
64	72	35	90
85	36	12	25

se obține tabloul

20	25	20
36	16	52
64	35	90
36	12	25

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int n, a[105][105];
    cin >> n;
    for(int i=0; i<n; i++)
        for(int j=0; j<n; j++)
            cin>>a[i][j];
    for(int i=0; i<n; i++)
    {
        int c=n-1-i; // coloana diagonalei secundare
        for(int j=c; j<n-1; j++)
            a[i][j]=a[i][j+1]; // deplasare spre stânga
    }
    // Se afișează doar primele n-1 coloane
    for(int i=0; i<n; i++)
    {
        for(int j=0; j<n-1; j++)
            cout<<a[i][j]<<" ";
        cout<<"\n";
    }
    return 0;
}
```

2.6 Subiect 2025. Varianta 4

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, m și n , și elementele unui tablou bidimensional cu m linii și n coloane, numere naturale distincte din intervalul $[0, 10^9]$. Programul modifică tabloul în memorie, înlocuind valorile tuturor elementelor aflate pe coloana care conține numărul minim memorat în tablou cu valoarea elementului aflat pe ultima linie și ultima coloană. Tabloul obținut este afișat pe ecran, câte o linie a tabloului pe câte o linie a ecranului, elementele fiecărei linii fiind separate prin câte un spațiu.

Exemplu: pentru $m=3$, $n=4$ și

17	12	5	19
10	16	2	4
11	21	8	9

se obține

17	12	9	19
10	16	9	4
11	21	9	9

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, a[102][102];
    cin >> m >> n;
    int minim, col;
    for(int i = 0; i < m; i++)
        for(int j = 0; j < n; j++)
        {
            cin >> a[i][j];
            if(i == 0 && j == 0)
            {
                minim = a[i][j];
                col = j;
            }
            else if(a[i][j] < minim)
            {
                minim = a[i][j];
                col = j;
            }
        }
    int val = a[m-1][n-1];
    for(int i = 0; i < m; i++)
        a[i][col] = val;
    for(int i = 0; i < m; i++)
    {
        for(int j = 0; j < n; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
    return 0;
}
```

2.7 Subiect 2025. Varianta 7

Parcarea unui mall are locuri de parcare dispuse pe nr rânduri, câte np pe fiecare rând, unul lângă altul. Trei prieteni vin cu câte o mașină la mall și caută un triplet de locuri libere alăturate, plasate toate trei doar pe primul sau toate trei doar pe ultimul rând al parării.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[3, 50]$, nr și np , reprezentând numărul de rânduri din parcare, respectiv numărul de locuri de pe fiecare rând, apoi $nr \cdot np$ valori din mulțimea $\{0, 1\}$, elemente ale unui tablou bidimensional cu nr linii și np coloane, reprezentând, în ordinea dispunerii lor pe rânduri, starea locurilor de parcare (0, pentru loc liber, sau 1, pentru loc ocupat).

Programul afișează pe ecran numărul de triplete de locuri libere pe care le pot găsi prietenii.

Exemplu: pentru $nr=4$, $np=5$ și tabloul alăturat, se afișează pe ecran valoarea 3 (pe prima linie pot găsi tripletul format din al doilea, al treilea și al patrulea loc, iar pe ultima linie pot găsi tripletul format din primul, al doilea și al treilea loc sau tripletul format din al doilea, al treilea, și al patrulea loc).

1	0	0	0	1
0	0	0	1	0
0	1	0	0	0
0	0	0	0	1

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int nr, np, a[50][50], cnt = 0;
    cin >> nr >> np;
    for(int i = 0; i < nr; i++)
        for(int j = 0; j < np; j++)
            cin >> a[i][j];
    // Verificăm primul rând
    for(int j = 0; j <= np - 3; j++)
        if(a[0][j] == 0 && a[0][j+1] == 0 && a[0][j+2] == 0)
            cnt++;
    // Verificăm ultimul rând
    for(int j = 0; j <= np - 3; j++)
        if(a[nr-1][j] == 0 && a[nr-1][j+1] == 0 && a[nr-1][j+2] == 0)
            cnt++;
    cout << cnt;
    return 0;
}
```


2.8 Subject 2025, Varianta 4. Sub I. 3

3. Indicați o declarație a unui tablou bidimensional **m**, care poate memora maximum 100 de numere reale.
- a. float m[2,50]; b. float m[4][25];
c. float m[10] x float m[10]; d. int m[100];

Răsp

uns corect: b.

A definit un tablou bidimensional cu 4 linii identificabile de la 0 la 3, a câte 25 de coloane identificabile de la 0 la 24. Total $4 \cdot 25 = 100$ elemente reale ce pot fi memorate.

2.9 Subject 2025, Varianta 6, Sub I. 4

4. Variabila **m** memorează elementele unui tablou bidimensional cu **100** de linii și **100** de coloane, numerotate de la **0** la **99**. Indicați o expresie C/C++ prin care poate fi accesat un element aflat pe diagonala principală a tabloului.
- a. **m[1,16]** b. **m[16][16]** c. **m(16,16)** d. **m(16)(1)**

Răspuns corect: b.

Pe diagonala principală indicele de linie și de coloană trebuie să coincidă.

2.10 Subject 2025. SIMULARE

3. Variabila `m` memorează elementele unui tablou bidimensional cu 2025 de linii și 2025 de coloane, numerotate de la 0 la 2024. Indicați expresia C/C++ prin care poate fi accesat un element aflat pe diagonala secundară a tabloului.
- a. `m[1999][25]` b. `m[52:1999]` c. `m[25][52]` d. `m[25:1999]`

Răspuns corect: b.

Pe diagonala secundară suma elementelor indicilor trebuie să fie egală 2024, numerotarea lor fiind de la 0 la 2024.

2.11 Subiect 2025. Varianta MODEL

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, k și n ($k \geq n$), și construiește în memorie un tablou bidimensional cu n linii și n coloane, astfel încât parcurgând diagonala principală, de sus în jos, să se obțină un șir strict crescător format din primii n multipli naturali nenuli ai lui k , și parcurgând fiecare linie a sa, de la stânga la dreapta, să se obțină câte un șir strict descrescător de numere naturale consecutive.

Programul afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.

Exemplu: pentru $k=5$ și $n=4$ se obține tabloul alăturat.

(10p.)

5	4	3	2
11	10	9	8
17	16	15	14
23	22	21	20

```
#include <iostream>
using namespace std;
int main()
{
    int n, k, a[100][10];
    cin >> k >> n;
    // Construim diagonala principală
    for(int i = 0; i < n; i++)
        a[i][i] = (i + 1) * k;
    // Completăm în stânga diagonalei
    for(int i = 0; i < n; i++)
        for(int j = i - 1; j >= 0; j--)
            a[i][j] = a[i][j + 1] + 1;
    // Completăm în dreapta diagonalei
    for(int i = 0; i < n; i++)
        for(int j = i + 1; j < n; j++)
            a[i][j] = a[i][j - 1] - 1;
    // Afisare
    for(int i = 0; i < n; i++)
    {
        for(int j = 0; j < n; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
    return 0;
}
```

Sau:

```

#include <iostream>
using namespace std;
int main()
{
    int k, n, a[100][100];
    cin >> k >> n;
    for(int i = 0; i < n; i++)
        for(int j = 0; j < n; j++)
            a[i][j] = (i + 1) * k + i - j;
    for(int i = 0; i < n; i++)
    {
        for(int j = 0; j < n; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
    return 0;
}

```

2.12 Subiect 2024. Varianta 8

O grădină este împărțită în parcele pătrate identice, dispuse pe rânduri succesive. Toate parcelele de pe un rând conțin același tip de plante (doar flori sau doar gazon) și oricare două rânduri alăturate conțin tipuri diferite de plante, **pe primul rând fiind gazon**. Pentru fiecare parcelă se memorează înălțimea acesteia, egală cu înălțimea medie a plantelor conținute, exprimată în decimetri. Pentru a pune în valoare florile, fiecare parcelă cu flori trebuie să fie mai înaltă cu cel puțin un decimetru decât oricare dintre parcelele cu gazon de pe rândul vecin anterior. S-a hotărât tunderea parcelelor cu gazon prea înalte, astfel încât ele să fie aduse la înălțimea maximă permisă.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, m și n , apoi elementele unui tablou bidimensional cu m linii și n coloane, numere naturale din intervalul $[2, 10]$, reprezentând înălțimile parcelelor, în ordinea dispunerii acestora în grădină, rând după rând, și pe fiecare rând de la stânga la dreapta. Programul modifică apoi tabloul în memorie corespunzător situației terenului după tunderea gazonului și afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.

Exemplu:

pentru $m=4$, $n=6$ și tabloul

6	2	3	7	9	4
7	8	5	9	8	5
8	7	2	6	2	9
7	2	5	9	3	8

se
obține
tabloul

4	2	3	4	4	4
7	8	5	9	8	5
1	1	1	1	1	1
7	2	5	9	3	8

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int m,n,a[105][105];
    cin>>m>>n;
    for(int i=0;i<m;i++)
        for(int j=0;j<n;j++)
            cin>>a[i][j];
    for(int i=0;i<m-1;i+=2) // rândurile cu gazon
    {
        int minim=a[i+1][0];
        for(int j=1;j<n;j++)
            if(a[i+1][j]<minim)
                minim=a[i+1][j];
        for(int j=0;j<n;j++)
            if(a[i][j]>=minim)
                a[i][j]=minim-1;
    }
    for(int i=0;i<m;i++)
    {
        for(int j=0;j<n;j++)
            cout<<a[i][j]<<" ";
        cout<<endl;
    }
    return 0;
}
```

2.13 Subiect 2024. Varianta 8. Sub II.3

Variabilele `i` și `j` sunt de tip întreg, iar variabila `a` memorează un tablou bidimensional cu 9 linii și 9 coloane, numerotate începând de la 0, având inițial toate elementele nule.

Scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie cu instrucțiuni adecvate, dintre care cel mult patru de atribuire, astfel încât, în urma executării secvenței obținute, variabila `a` să memoreze tabloul alăturat.

```
for (i=0; i<9; i++)  
    for (j=0; j<9; j++)
```

.....

(6p.)

4	4	4	4	2	2	2	2	2
4	4	4	2	2	2	2	2	2
4	4	2	2	2	2	2	2	2
4	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2	4
2	2	2	2	2	2	2	4	4
2	2	2	2	2	2	4	4	4
2	2	2	2	2	4	4	4	4

```
for (int i=0; i<9; i++)  
    for (int j=0; j<9; j++)  
        if (i+j<4 || i+j>12)  
            a[i][j]=4;  
        else  
            a[i][j]=2;
```

2.14 Subiect 2024. Varianta 4

Fiind date două numere naturale **a** și **b**, numim valoare **generată** de **a** și **b** un număr obținut din **a** prin alipirea la stânga sau la dreapta sa a cifrelor lui **b**, în ordinea în care apar în acesta.

Exemplu: dacă **a=123** și **b=45**, se pot obține două numere generate de acestea: **12345** și **45123**.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul **[2, 50]**, **m** și **n**, și construiește în memorie un tablou bidimensional cu **m** linii, numerotate de la **1** la **m**, și **n** coloane, numerotate de la **1** la **n**, în care fiecare element este egal cu cea mai mică valoare generată de numerele de ordine ale liniei, respectiv coloanei pe care se află.

Programul afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele fiecărei linii fiind separate prin câte un spațiu.

Exemplu: pentru **m=5** și **n=4** se obține tabloul alăturat.

(10p.)	11	12	13	14
	12	22	23	24
	13	23	33	34
	14	24	34	44
	15	25	35	45

```
#include <iostream>
using namespace std;
int main()
{
    int m,n,a[51][51];
    cin>>m>>n;
    for(int i=1; i<=m; i++)
        for(int j=1; j<=n; j++)
        {
            int x,y;
            if(j<10)
                x=i*10+j;
            else
                x=i*100+j;
            if(i<10)
                y=j*10+i;
            else
                y=j*100+i;
            if(x<y)
                a[i][j]=x;
            else
                a[i][j]=y;
        }
    for(int i=1; i<=m; i++)
    {
        for(int j=1; j<=n; j++)
            cout<<a[i][j]<<" ";
        cout<<endl;
    }
    return 0;
}
```

2.15 Subiect 2024. Varianta SIMULARE. Sub. II.3

Variabilele `i` și `j` sunt de tip întreg, iar variabila `a` memorează un tablou bidimensional cu 5 linii și 5 coloane, numerotate începând de la 0, având inițial toate elementele egale cu 1.

Scrieți secvența de instrucțiuni C/C++ de mai jos, înlocuind punctele de suspensie cu instrucțiuni adecvate, dintre care **cel mult patru de atribuire**, astfel încât, în urma executării secvenței obținute, variabila `a` să memoreze tabloul alăturat.

```
for (i=0; i<5; i++)  
    for (j=0; j<5; j++)  
        .....
```

(6p.)

0	0	0	0	0
0	2	2	2	2
0	2	4	4	4
0	2	4	6	6
0	2	4	6	8

```
for (i=0; i<5; i++)  
    for (j=0; j<5; j++)  
        if (i<j)  
            a[i][j]=2*i;  
        else  
            a[i][j]=2*j;
```

2.16 Subiect 2023. Varianta 7

Un joc folosește o tablă dreptunghiulară, pe care sunt reprezentate celule de dimensiune egală, dispuse pe m linii și pe n coloane. În fiecare celulă este înscris un număr natural.

Numim **pătrat de valoare p** patru celule ale tablei, situate pe două linii consecutive și pe două coloane consecutive, cu proprietatea că suma valorilor înscrise în aceste celule este egală cu p . Scopul jocului este determinarea unui pătrat de valoare maximă.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 20]$, m și n , apoi elementele unui tablou bidimensional cu m linii și n coloane, numere naturale din intervalul $[0, 10^4]$, reprezentând numerele înscrise pe tabla de joc, în ordinea dispunerii celulelor corespunzătoare. Programul determină un pătrat de valoare maximă al tablei și afișează pe ecran această valoare.

1	1	2	30
3	10	1	2
1	13	12	1
1	2	3	1
7	1	20	1

Exemplu: pentru $m=5$, $n=4$ și tabloul alăturat, se afișează pe ecran valoarea 36, corespunzătoare pătratului evidențiat în figură. (10p.)

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

int main()
{
    int m, n, a[21][21];
    cin >> m >> n;
    for(int i = 0; i < m; i++)
        for(int j = 0; j < n; j++)
            cin >> a[i][j];
    int maxim = 0;
    for(int i = 0; i < m - 1; i++)
        for(int j = 0; j < n - 1; j++)
        {
            int s = a[i][j] + a[i][j+1]
                    + a[i+1][j] + a[i+1][j+1];
            if(s > maxim)
                maxim = s;
        }
    cout << maxim;
    return 0;
}
```


2.17 Subiect 2023. Varianta 5

Pentru a identifica punctele în care se concentrează apa în albia unui râu în cazul secetei, se determină talvegul acesteia – linia care unește punctele cele mai adânci ale albiei. În acest scop s-au stabilit **ns** secțiuni transversale pe cursul apei, numerotate începând de la 1, și în cadrul fiecărei secțiuni s-a măsurat adâncimea apei în **np** puncte, numerotate începând de la 1. Din fiecare secțiune, în ordine, se include în talveg cel mai adânc punct al acesteia, iar dacă în secțiune sunt mai multe puncte aflate la aceeași adâncime, maximă, se va lua în considerare doar primul dintre ele, ca în exemplu.

Scrieți un program C/C++ care citește de la tastatură două numere naturale, **ns** și **np** ($ns \in [1, 10^3]$, $np \in [1, 50]$), și cele **ns · np** elemente ale unui tablou bidimensional, valori naturale din intervalul $[0, 10^2]$.

Fiecare linie a tabloului corespunde câte unei secțiuni, în ordinea numerotării acestora, iar valorile memorate pe linie reprezintă adâncimile celor **np** puncte stabilite pentru acea secțiune, în ordinea numerotării lor. Programul afișează pe ecran, pentru fiecare secțiune, o pereche formată din numărul de ordine al secțiunii și numărul de ordine al punctului său care s-a inclus în talveg. Numerele din fiecare pereche sunt afișate separate prin câte un caracter : (două puncte), iar fiecare pereche este urmată de un spațiu.

Exemplu: pentru **ns=6**, **np=4** și tabloul alăturat, se afișează pe ecran valorile:

1:3 2:2 3:2 4:2 5:4 6:3

(10p.)

2	4	5	3
2	6	6	3
1	5	2	5
1	3	3	3
3	4	3	5
0	1	2	1

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

int main() {
    int ns, np;
    cin >> ns >> np;
    int a[1005][55];
    /// Citirea matricei
    for (int i = 1; i <= ns; i++)
        for (int j = 1; j <= np; j++)
            cin >> a[i][j];
    /// Parcurgerea fiecărei linii
    for (int i = 1; i <= ns; i++) {
        int maxim = a[i][1];
        int poz = 1;
        for (int j = 2; j <= np; j++)
            if (a[i][j] > maxim) {
                maxim = a[i][j];
                poz = j;
            }
        cout << i << ":" << poz << " ";
    }
    return 0;
}
```


2.18 Subiect 2023. Varianta 6

La o expoziție sunt prezentate mai multe tipuri de flori, fiecare având un anumit cod. Zona de expunere este organizată pe n rânduri, fiecare având câte n parcele de dimensiune egală. Pe primul rând sunt expuse n tipuri de flori distincte, fiecare parcelă conținând flori de un anumit tip. Celelalte parcele sunt ocupate astfel:

- toate parcelele aflate în dreapta diagonalei secundare a zonei de expunere sunt ocupate de gazon;
- fiecare parcelă, aflată pe diagonala secundară a zonei de expunere sau în stânga acestei diagonale, conține flori cu un cod de cel mult două cifre, obținut din ultimele două cifre ale sumei codurilor florilor aflate în două parcele de pe rândul anterior, și anume cea de pe aceeași coloană, respectiv cea de pe coloana din dreapta sa; dacă prima dintre cifrele codului obținut este 0, ea este ignorată, ca în exemplu.

Scrieți un program C/C++ care citește de la tastatură un număr natural, n ($n \in [2, 20]$), apoi n numere naturale distincte din intervalul $[0, 10^2)$, reprezentând, în această ordine, de la stânga la dreapta, codurile florilor din parcelele aflate pe primul rând al zonei de expunere; gazonul are codul -1. Programul construiește în memorie un tablou bidimensional cu n linii și n coloane, în care fiecare linie corespunde unui rând de parcele și fiecare element al liniei corespunde unei parcele, în ordinea acestora, și memorează codul plantelor (flori sau gazon) plasate în acea parcelă.

Programul afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.

Exemplu: dacă se citesc numerele $n=5$, apoi 90, 10, 4, 27, 30 se obține tabloul alăturat.

90	10	4	27	30
0	14	31	57	-1
14	45	88	-1	-1
59	33	-1	-1	-1
92	-1	-1	-1	-1

(10p.)

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

int main() {
    int n;
    cin >> n;
    int a[21][21];
    /// Citirea primei linii
    for (int j = 1; j <= n; j++)
        cin >> a[1][j];
    /// Construirea matricei
    for (int i = 2; i <= n; i++) {
        for (int j = 1; j <= n; j++) {
            if (i + j <= n + 1)
                a[i][j] = (a[i - 1][j] + a[i - 1][j + 1]) % 100;
            else
                a[i][j] = -1;
        }
    }
    /// Afișarea matricei
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```

2.19 Subiect 2023. Varianta SIMULARE. Sub II.3

Variabilele **i** și **j** sunt de tip întreg, iar variabila **a** memorează un tablou bidimensional cu 5 linii și 5 coloane, numerotate de la 1 la 5, având inițial toate elementele nule.

Fără a utiliza alte variabile decât cele menționate, scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila **a** să memoreze tabloul alăturat.

```
for (i=1; i<=5; i++)  
    for (j=1; j<=5; j++)
```

.....

(6p.)

4	3	2	1	0
4	3	2	1	1
4	3	2	2	2
4	3	3	3	3
4	4	4	4	4

```
for (i=1; i<=5; i++)  
    for (j=1; j<=5; j++)  
        if (i-1>5-j)  
            a[i][j]=i-1;  
        else  
            a[i][j]=5-j;
```

2.20 Subiect 2023. Varianta MODEL

Suprafața unui loc de joacă pentru copii este împărțită în $n \cdot n$ zone de formă pătrată dispuse pe n rânduri și n coloane, numerotate de la 1 la n , în fiecare zonă fiind plasat un tobogan, un leagăn, un balansoar sau un carusel. Zonelor li se atribuie câte un cod, în funcție de destinație: codul 0 pentru tobogan, codul 1 pentru leagăn, codul 2 pentru balansoar și codul 3 pentru carusel.

Scrieți un program C/C++ care citește de la tastatură un număr natural, n ($n \in [1, 10^2]$), și cele $n \cdot n$ elemente ale unui tablou bidimensional, reprezentând codurile corespunzătoare zonelor din locul de joacă, în ordinea parcurgerii lor rând după rând, și a fiecărui rând de la stânga la dreapta.

Programul determină zonele destinate plasării câte unui carusel, care NU au nicio latură comună cu o zonă cu tobogan și, pentru fiecare dintre acestea, afișează pe ecran numărul de ordine al rândului pe care se află. Numerele afișate sunt separate prin câte un spațiu, iar dacă nu există nicio astfel de zonă, se afișează pe ecran mesajul **nu exista**.

Exemplu: pentru $n=6$ și tabloul alăturat se afișează pe ecran, nu neapărat în această ordine, numerele: 2 3 3 6. (10p.)

2	0	0	2	1	3
3	1	3	1	2	0
2	1	3	3	2	0
0	2	1	3	1	1
3	1	2	0	0	2
0	0	0	2	1	3

```
#include <iostream>
using namespace std;
int a[105][105];
int dr[] = {-1, 1, 0, 0};
int dc[] = {0, 0, -1, 1};
int main() {
    int n;
    cin >> n;
    /// Bordarea matricei cu -1
    for (int i = 0; i <= n + 1; i++) {
        a[0][i] = -1;
        a[n + 1][i] = -1;
        a[i][0] = -1;
        a[i][n + 1] = -1;
    }
    /// Citirea matricei
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n; j++)
            cin >> a[i][j];
    bool exista = 0;
    /// Parcurgerea matricei
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n; j++)
            if (a[i][j] == 3) {
                bool ok = 1;
                for (int k = 0; k < 4; k++) {
                    int lin = i + dr[k];
                    int col = j + dc[k];
                    if (a[lin][col] == 0) {
                        ok = 0;
                        break;
                    }
                }
                if (ok == 1) {
                    cout << i << " ";
                    exista = 1;
                }
            }
    if (exista == 0)
        cout << "nu exista";
    return 0;
}
```

2.21 Subiect 2022. Varianta 1

Pentru prevenirea eroziunii la Marea Neagră, autoritățile au hotărât înnisiparea unei plaje. Plaja a fost împărțită în parcele de formă pătrată cu latura de 1 metru, dispuse ca elementele unui tablou bidimensional, reținându-se nivelul fiecărei parcele, măsurat în metri, de la nivelul mării. Înnisiparea se realizează doar pe acele parcele care au nivelul strict mai mic decât al celei mai joase dintre parcelele vecine cu ea, adăugând nisip, astfel încât cele două să aibă același nivel. Două parcele sunt vecine dacă au o latură comună.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, m și n , apoi elementele unui tablou bidimensional cu m linii și n coloane, numere naturale din intervalul $[1, 10]$, reprezentând nivelurile parcelelor, în ordinea dispunerii acestora pe plajă.

Programul afișează pe ecran cantitatea totală de nisip adăugată, măsurată în metri cubi.

Exemplu: pentru $m=5$, $n=4$ și tabloul alăturat, se vor înnisipa parcelele marcate, pentru a fi aduse la nivelul parcelelor vecine corespunzătoare, și se va afișa valoarea 7 ($1+3+1+2$).

5	3	4	6
7	5	9	3
6	8	3	9
4	5	2	3
3	1	4	4

(10p.)

```
#include <iostream>
using namespace std;
int a[105][105];
int dr[] = {-1, 1, 0, 0};
int dc[] = {0, 0, -1, 1};
int main()
{
    int m, n;
    cin >> m >> n;
    for (int i = 1; i <= m; i++)
        for (int j = 1; j <= n; j++)
            cin >> a[i][j];

    int total = 0;
    for (int i = 1; i <= m; i++)
        for (int j = 1; j <= n; j++)
        {
            int minim = 11;    /// nivelurile sunt între 1 și 10
            for (int k = 0; k < 4; k++)
            {
                int lin = i + dr[k];
                int col = j + dc[k];
                if (lin >= 1 && lin <= m && col >= 1 && col <= n)
                    if (a[lin][col] < minim)
                        minim = a[lin][col];
            }
            if (a[i][j] < minim)
                total += minim - a[i][j];
        }
    cout << total;
    return 0;
}
```


2.22 Subiect 2022. Varianta 4

Suprafața unei scene a fost împărțită în zone dispuse ca elementele unui tablou bidimensional cu număr egal de linii și coloane. Fiecare instrument are alocată o zonă și este identificat printr-un număr natural. Dirijorul hotărăște eliminarea tuturor instrumentelor plasate pe diagonala principală a tabloului și rearanjarea celor păstrate, prin deplasarea unora dintre acestea cu câte o poziție spre stânga, astfel încât să nu existe zone intermediare libere, ca în exemplu.

Scrieți un program C/C++ care citește de la tastatură numere naturale din intervalul $[2, 10^2]$, și anume n și elementele unui tablou bidimensional cu n linii și n coloane, reprezentând numerele de identificare ale instrumentelor, în ordinea plasării lor inițiale pe scenă. Programul modifică apoi tabloul în memorie corespunzător hotărârii dirijorului și afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.

Exemplu: pentru $n=4$ și tabloul

20	30	80	52
71	93	64	51
62	81	55	49
90	65	32	22

se obține tabloul

30	80	52
71	64	51
62	81	49
90	65	32

(10p.)

```
#include <iostream>
using namespace std;
int main() {
    int n;
    cin >> n;
    int a[105][105];
    /// Citirea matricei
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n; j++)
            cin >> a[i][j];
    /// Eliminarea diagonalei principale
    for (int i = 1; i <= n; i++)
        for (int j = i; j < n; j++)
            a[i][j] = a[i][j + 1];
    /// Afișarea
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= n - 1; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```

2.23 Subiect 2022. Varianta SIMULARE. Sub. II.3

3. Într-o seră se folosesc vase pătrate, care conțin câte o singură floare (lalea, frezie sau narcisă). Vasele sunt așezate pe trei rânduri și sunt aliniate ca în exemplu. Patru flori de același tip formează un **romb** dacă vasele în care se află sunt dispuse în jurul unui alt vas, cu care fiecare are în comun câte o latură. Variabilele `j` și `ok` sunt întregi, iar variabila `a` este un tablou bidimensional cu 3 linii și 15 coloane, cu elemente de tip `char`, în care fiecare linie, în ordine, memorează succesiunea de flori aflate pe un rând al serei, notându-se cu **L** lalelele, cu **F** freziile și cu **N** narcisele. Fiecare linie a tabloului corespunde, în ordine, unui rând de vase din seră. Liniile și coloanele sunt numerotate începând de la 0. Scrieți secvența C/C++ de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila `ok` să aibă valoarea 1 dacă florile formează cel puțin un romb, sau valoarea 0 în caz contrar. **Exemplu:** în urma executării secvenței pentru tabloul de mai jos, `ok=1` (sunt evidențiate 4 romburi).

```
ok=.....;
for (j=.....;j<.....;j++)
    .....
```

L	L	L	N	L	F	F	F	N	F	N	F	N	F	F
N	L	L	L	N	L	L	F	F	F	F	N	F	N	F
F	N	L	F	L	N	F	L	F	F	N	L	F	F	N

(6p.)

```
ok=0;
for (j=1; j<14 && ok==0; j++)
    if (a[0][j]==a[1][j-1] && a[0][j]==a[1][j+1] && a[0][j]==a[2][j])
        ok=1;
```

2.24 Subiect 2021. Varianta 4

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, k și n , și construiește în memorie un tablou bidimensional cu n linii și n coloane, astfel încât parcurgând diagonala principală, de sus în jos, să se obțină un șir strict crescător format din primii n multipli naturali nenuli ai lui k și parcurgând fiecare linie a sa, de la stânga la dreapta, să se obțină câte un șir strict crescător de numere naturale consecutive. Programul afișează pe ecran tabloul obținut, fiecare linie a tabloului pe câte o linie a ecranului, elementele de pe aceeași linie fiind separate prin câte un spațiu.

Exemplu: pentru $k=3$ și $n=4$ se obține tabloul de mai jos

3	4	5	6
5	6	7	8
7	8	9	10
9	10	11	12

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int k, n;
    cin >> k >> n;
    int a[101][101];
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n; j++)
            a[i][j] = i * k + (j - i);
    for (int i = 1; i <= n; i++)
    {
        for (int j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```

2.25 Subiect 2021. Varianta 7

Scrieți un program C/C++ care citește de la tastatură numere naturale: n ($n \in [2, 20]$), k ($k \in [2, n]$) și $n \cdot n$ numere din intervalul $[0, 10^9]$, elemente ale unui tablou bidimensional cu n linii și n coloane. Programul transformă tabloul în memorie, interschimbând șirul elementelor situate pe linia a k -a, în stânga diagonalei principale, parcurse de la stânga la dreapta, cu șirul elementelor situate pe coloana a k -a, deasupra diagonalei principale, parcurse de sus în jos, ca în exemplu. Elementele tabloului obținut sunt afișate pe ecran, fiecare linie pe câte o linie a ecranului, cu elementele fiecărei linii separate prin câte un spațiu.

Exemplu: pentru $n=5$,
 $k=4$ și tabloul

2	4	3	5	6
8	0	9	8	7
2	6	9	0	5
6	1	3	6	9
7	3	9	4	2

se obține tabloul

2	4	3	6	6
8	0	9	1	7
2	6	9	3	5
5	8	0	6	9
7	3	9	4	2

(10p.)

```

#include <iostream>
using namespace std;
int main() {
    int n, k;
    cin >> n >> k;
    int a[21][21];
    /// Citirea matricei
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n; j++)
            cin >> a[i][j];
    /// Interschimbarea elementelor
    for (int i = 1; i < k; i++)
        swap(a[k][i], a[i][k]);
    /// Afisare matrice
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}

```


2.26 Subiect 2021. Varianta SIMULARE

Scrieți un program C/C++ care citește de la tastatură numere naturale, în această ordine: n ($n \in [2, 20]$), apoi $n \cdot n$ numere din intervalul $[0, 10^2]$, elemente ale unui tablou bidimensional cu n linii și n coloane, iar la final valoarea k ($k \in [2, n]$); programul transformă tabloul în memorie, prin deplasarea circulară spre dreapta, cu câte o poziție, a tuturor valorilor aflate pe linia a k -a, în stânga diagonalei principale, ca în exemplu. Elementele tabloului obținut sunt afișate pe ecran, linie cu linie, fiecare linie pe câte o linie a ecranului, cu elementele fiecărei linii separate prin câte un spațiu.

Exemplu: pentru $n=6, k=5$ și tabloul

0	1	4	5	3	2
0	6	0	7	5	0
4	0	0	6	0	6
2	0	0	5	0	0
3	0	6	4	0	8
0	6	9	0	0	0

se obține tabloul

0	1	4	5	3	2
0	6	0	7	5	0
4	0	0	6	0	6
2	0	0	5	0	0
4	3	0	6	0	8
0	6	9	0	0	0

(10p.)

```
#include <iostream>
using namespace std;
int main() {
    int n, k;
    cin >> n;
    int a[21][21];
    /// Citirea matricei
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n; j++)
            cin >> a[i][j];
    cin >> k;
    /// Deplasare circulara spre dreapta
    int aux = a[k][k - 1];
    for (int j = k - 1; j >= 2; j--)
        a[k][j] = a[k][j - 1];
    a[k][1] = aux;
    /// Afisare matrice
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```

2.27 Subiect 2021. Varianta TEST ANTRENAMENT 1

Scrieți un program C/C++ care citește de la tastatură numărul natural n ($n \in [2, 10^2]$) și elementele unui tablou bidimensional cu n linii și n coloane, numere naturale din intervalul $[0, 10^9]$.

Programul afișează pe ecran, separate prin câte un spațiu, elementele primului pătrat concentric, parcurs în sens invers al acelor de ceasornic, începând din colțul său stângasus, ca în exemplu. Primul pătrat concentric este format din prima și ultima linie, prima și ultima coloană a tabloului.

Exemplu: pentru $n=5$ și tabloul alăturat, se afișează pe ecran numerele

1 2 3 4 5 6 7 8 9 0 2 4 6 8 1 3

(10p.)

1	3	1	8	6
2	9	2	7	4
3	5	8	5	2
4	1	6	3	0
5	6	7	8	9

```
#include <iostream>
using namespace std;
int main() {
    int n;
    cin >> n;
    int a[101][101];
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            cin >> a[i][j];
    /// Prima coloana
    for (int i = 0; i < n; i++)
        cout << a[i][0] << " ";
    /// Ultima linie
    for (int j = 1; j < n; j++)
        cout << a[n - 1][j] << " ";
    ///Ultima coloana
    for (int i = n - 2; i >= 0; i--)
        cout << a[i][n - 1] << " ";
    /// Prima linie
    for (int j = n - 2; j >= 1; j--)
        cout << a[0][j] << " ";
    return 0;
}
```

2.28 Subiect 2021. Varianta TEST ANTRENAMENT 2

Scrieți un program C/C++ care citește de la tastatură un număr natural, n ($n \in [2, 20]$), și construiește în memorie un tablou bidimensional cu n linii și n coloane, astfel încât fiecare linie a sa are proprietățile:

- elementul situat pe diagonală secundară are valoarea n ;
- începând de la diagonală secundară, spre stânga, elementele formează un șir strict descrescător de numere consecutive, iar începând de la diagonală secundară, spre dreapta, elementele formează tot un șir strict descrescător de numere consecutive.

Programul afișează pe ecran tabloul construit, fiecare linie a tabloului pe câte o linie a ecranului, cu elementele aflate pe aceeași linie separate prin câte un spațiu.

Exemplu: dacă $n=5$ se afișează pe ecran tabloul alăturat.

(10p.)

1	2	3	4	5
2	3	4	5	4
3	4	5	4	3
4	5	4	3	2
5	4	3	2	1

Observăm că:

- pe **diagonală secundară** ($i+j = n-1$) se află valoarea n ;
- spre stânga și spre dreapta valorile scad cu 1 la fiecare pas.

Dacă elementul (i,j) este la distanța $d = |i+j - (n-1)|$ față de diagonală secundară, atunci:

$$a[i][j] = n - |i+j - (n-1)|$$

```
#include <iostream>
#include <cmath>
using namespace std;
int main()
{
    int n, a[21][21];
    cin >> n;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            a[i][j] = n - abs(i + j - (n - 1));
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```

2.29 Subiect 2021. Varianta TEST ANTRENAMENT 3

O sală de spectacole are mai multe rânduri de scaune, toate rândurile având același număr de locuri. O persoană poate rezerva mai multe locuri, dar cel mult câte unul pe un rând. Configurația sălii este memorată ca un tablou bidimensional, în care prima **coloană** corespunde primului rând de scaune, iar ultima **coloană** corespunde ultimului rând de scaune. Locurile libere sunt notate cu 0, iar cele rezervate sunt notate cu numere naturale nenule, astfel încât toate locurile rezervate de o persoană sunt notate cu același număr, iar cele rezervate de persoane distincte sunt notate cu numere distincte.

Scrieți un program C/C++ care citește de la tastatură numerele naturale din intervalul $[2, 20]$, m și n , apoi $m \cdot n$ numere naturale din intervalul $[0, 400]$, elemente ale tabloului bidimensional cu m linii și n coloane, reprezentând configurația sălii. Programul verifică dacă există persoane care au rezervat locuri atât pe primul, cât și pe ultimul rând al sălii, și afișează pe ecran numerele cu care sunt notate aceste locuri, ca în exemplu.

Numerele sunt afișate într-o ordine oarecare, separate prin câte un spațiu, iar dacă nu există astfel de locuri se afișează pe ecran mesajul **nu exista**.

Exemplu: pentru $m=5$, $n=4$ și tabloul alăturat se afișează pe ecran, nu neapărat în această ordine, numerele 6 8

3	7	0	0
0	0	5	8
6	2	0	1
0	3	0	6
8	0	6	4

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, a[21][21];
    cin >> m >> n;
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            cin >> a[i][j];
    bool ok = 0;
    for (int i = 0; i < m; i++)
        if (a[i][0] != 0) cautarea pe coloana 0
        {
            for (int k = 0; k < m; k++)
                if (a[k][n-1] == a[i][0]) cautarea pe col n-1
                {
                    cout << a[i][0] << " ";
                    ok = 1;
                    break;
                }
        }
    if (ok == 0)
        cout << "nu exista";
    return 0;
}
```

2.30 Subiect 2021. Varianta TEST ANTRENAMENT 4

Scrieți un program C/C++ care citește de la tastatură numere naturale din intervalul $[2, 10^2]$: m , n și elementele unui tablou bidimensional cu m linii și n coloane. Programul afișează pe ecran, pentru fiecare linie a sa, cea mai mare dintre valorile strict mai mici decât 21 memorate în aceasta, sau mesajul `nu exista`, dacă nu există nicio astfel de valoare pe linia respectivă. Numerele, respectiv mesajele, sunt afișate pe linii separate ale ecranului, în ordinea liniilor corespunzătoare din tablou.

Exemplu: pentru $m=3$, $n=5$ și tabloul alăturat se afișează pe ecran

16

nu exista

9

	6	16	21	4	90
	92	26	36	95	80
	5	2	9	7	3

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, a[101][101];
    cin >> m >> n;
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            cin >> a[i][j];
    for (int i = 0; i < m; i++)
    {
        int maxl = -1;
        for (int j = 0; j < n; j++)
            if (a[i][j] < 21 && a[i][j] > maxl)
                maxl = a[i][j];
        if (maxl == -1)
            cout << "nu exista\n";
        else
            cout << maxl << "\n";
    }
    return 0;
}
```

2.31 Subiect 2021. Varianta TEST ANTRENAMENT 5

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, m și n , și construiește în memorie un tablou bidimensional cu m linii și n coloane, numerotate începând cu 1, astfel încât elementul de pe linia i și coloana j primește ca valoare ultima cifră a produsului $i \cdot j$.

Programul afișează pe ecran elementele tabloului obținut, linie cu linie, fiecare linie a tabloului pe câte o linie a ecranului, cu valorile aflate pe aceeași linie separate prin câte un spațiu.

Exemplu: dacă $m=4$ și $n=5$ se afișează pe ecran tabloul alăturat.

(10p.)

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

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, a[101][101];
    cin >> m >> n;
    /// Construirea matricei în memorie cu indicii de la 1
    for (int i = 1; i <= m; i++)
        for (int j = 1; j <= n; j++)
            a[i][j] = (i * j) % 10;
    /// Afișarea matricei
    for (int i = 1; i <= m; i++)
    {
        for (int j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << "\n";
    }
    return 0;
}
```


2.32 Subiect 2021. Varianta TEST ANTRENAMENT 6

Scrieți un program C/C++ care citește de la tastatură un număr natural, n ($n \in [3, 20]$), și construiește în memorie un tablou bidimensional cu n linii și n coloane, având proprietățile:

- toate elementele situate pe diagonala secundară sunt nule;
- prima linie conține un șir strict descrescător de numere consecutive, iar ultima linie conține un șir strict crescător de numere consecutive;
- fiecare dintre celelalte linii conține, începând cu prima poziție, până la diagonala secundară inclusiv, de la stânga la dreapta, un șir strict descrescător de numere consecutive, iar începând de la diagonala secundară, inclusiv, până la ultima poziție, de la stânga la dreapta, un șir strict crescător de numere consecutive.

Programul afișează pe ecran tabloul construit, fiecare linie a tabloului pe câte o linie a ecranului, cu elementele aflate pe aceeași linie separate prin câte un spațiu.

Exemplu: dacă $n=4$ se afișează pe ecran tabloul alăturat.

(10p.)

3	2	1	0
2	1	0	1
1	0	1	2
0	1	2	3

```
#include <iostream>
using namespace std;
int main()
{
    int n, a[21][21];
    cin >> n;
    for (int i = 0; i < n; i++)
        a[i][n-1-i] = 0;
    /*Se calculeaza elementele deasupra diagonalei sec.,
    pe fiecare linie de la coloana diagonalei secundare spre stanga
    si simetricul sau.*/
    for (int i = 0; i < n-1; i++)
        for (int j = n-2-i; j >= 0; j--)
        {
            a[i][j] = a[i][j+1] + 1;
            a[n-j-1][n-i-1] = a[i][j];
        }
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```


2.33 Subiect 2021. Varianta TEST ANTRENAMENT 7

Scrieți un program C/C++ care citește de la tastatură un număr natural, x ($x \in [1, 10^9]$), și construiește în memorie un tablou bidimensional, pentru care atât numărul de linii, cât și numărul de coloane sunt egale cu numărul de cifre ale lui x , iar elementele fiecărei linii au ca valori cifrele lui x , în ordine, ca în exemplu.

Elementele tabloului obținut sunt afișate pe ecran, linie cu linie, fiecare linie a tabloului pe câte o linie a ecranului, cu elementele de pe aceeași linie separate prin câte un spațiu.

Exemplu: dacă $x=1359$, se afișează tabloul alăturat.

(10p.)

1	3	5	9
1	3	5	9
1	3	5	9
1	3	5	9

```
#include <iostream>
using namespace std;
int main()
{
    int x, y;
    int n = 0, a[11][11];
    cin >> x;
    y = x;
    // Determinarea numărului de cifre
    while (y)
    {
        n++;
        y /= 10;
    }
    /* Memorarea cifrelor de la dreapta la stanga
    incepand cu ultima coloana n-1, n-2, ..., 1*/
    y = x;
    for (int j = n - 1; j >= 0; j--)
    {
        a[0][j] = y % 10;
        y /= 10;
    }
    // Construirea restului de matrice prin copierea liniei 0
    for (int i = 1; i < n; i++)
        for (int j = 0; j < n; j++)
            a[i][j] = a[0][j];
    // Afișarea matricei
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```

2.34 Subiect 2021. Varianta TEST ANTRENAMENT 8

Scrieți un program C/C++ care citește de la tastatură numărul natural n ($n \in [5, 50]$) și elementele unui tablou bidimensional cu n linii și n coloane, numere naturale din intervalul $[0, 10^2]$. Programul afișează pe ecran suma numerelor din zona delimitată de cele două diagonale și ultima coloană a tabloului, ca în exemplu.

Exemplu: pentru $n=7$ și tabloul alăturat, se afișează pe ecran 12.

(10p.)

1	2	3	4	5	5	6
7	8	9	0	3	1	2
4	6	8	0	1	1	3
8	6	3	6	2	4	7
5	7	9	2	2	5	8
1	4	7	0	5	3	6
9	2	5	8	5	9	1

```
#include <iostream>
using namespace std;
int main()
{
    int n, a[51][51];
    cin >> n;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            cin >> a[i][j];
    int s = 0;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n-1; j++)
            if (j > i && j > n - 1 - i)
                s += a[i][j];
    cout << s;
    return 0;
}
```

2.35 Subiect 2021. Varianta TEST ANTRENAMENT 9

Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 4 linii și 5 coloane, numerotate începând de la 0, cu elemente numere întregi, inițial toate nule. Fără a utiliza alte variabile decât cele menționate, scrieți secvența de mai jos astfel încât, în urma executării secvenței obținute, variabila a să memoreze tabloul alăturat.

```
for(i=0; i<4; i++)
    for(j=0; j<5; j++)
        .....
```

(6p.)

0	1	2	0	1
1	2	0	1	2
2	0	1	2	0
0	1	2	0	1

```
for(i=0; i<4; i++)
    for(j=0; j<5; j++)
        a[i][j] = (i+j) % 3;
```

2.36 Subiect 2021. Varianta TEST ANTRENAMENT 10

O valoare **filtrează** două șiruri dacă există doi termeni care au acea valoare, unul fiind în primul șir, iar celălalt în al doilea șir.

Scrieți un program C/C++ care citește de la tastatură numere naturale din intervalul $[2, 20]$: m , n și elementele unui tablou bidimensional cu m linii și n coloane, cu proprietatea că nu există două elemente egale situate pe aceeași linie sau pe aceeași coloană.

Programul afișează pe ecran valorile care pot filtra șirul format din primele $n-1$ elemente de pe prima linie, respectiv șirul format din ultimele $m-1$ elemente ale ultimei coloane a tabloului, ca în exemplu. Valorile sunt afișate într-o ordine oarecare, separate prin câte un spațiu, sau mesajul `nu exista`, dacă nu există astfel de valori.

Exemplu: pentru $m=5$, $n=4$ și tabloul alăturat, se afișează pe ecran, nu neapărat în această ordine, numerele 4 7

4	5	7	2
2	7	3	6
7	6	4	0
6	9	8	7
8	0	5	4

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, a[21][21];
    cin >> m >> n;
    for (int i = 0; i < m; i++)
        for (int j = 0; j < n; j++)
            cin >> a[i][j];
    bool ok = 0;
    for (int j = 0; j < n - 1; j++)
        for (int i = 1; i < m; i++)
            if (a[0][j] == a[i][n - 1])
            {
                cout << a[0][j] << " ";
                ok = 1;
                break;
            }
    if (ok == 0)
        cout << "nu exista";
    return 0;
}
```

2.37 Subiect 2021. Varianta TEST ANTRENAMENT 11

Scrieți un program C/C++ care citește de la tastatură numere naturale: n ($n \in [2, 20]$), k ($k \in [2, n]$) și $n \cdot n$ numere din intervalul $[0, 10^9]$, elemente ale unui tablou bidimensional cu n linii și n coloane. Programul transformă tabloul în memorie, interschimbând șirul elementelor situate pe linia a k -a, în stânga diagonalei principale, parcurse de la stânga la dreapta, cu șirul elementelor situate pe coloana a k -a, deasupra diagonalei principale, parcurse de sus în jos, ca în exemplu. Elementele tabloului obținut sunt afișate pe ecran, fiecare linie pe câte o linie a ecranului, cu elementele fiecărei linii separate prin câte un spațiu.

Exemplu: pentru $n=5$, $k=4$ și tabloul

2	4	3	5	6
8	0	9	8	7
2	6	9	0	5
6	1	3	6	9
7	3	9	4	2

se obține tabloul

2	4	3	6	6
8	0	9	1	7
2	6	9	3	5
5	8	0	6	9
7	3	9	4	2

(10p.)

```
#include <iostream>
using namespace std;
int main()
{
    int n, k, a[21][21];
    cin >> n >> k;
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n; j++)
            cin >> a[i][j];
    for (int i = 1; i <= k; i++)
        swap(a[k][i], a[i][k]);
    for (int i = 1; i <= n; i++)
    {
        for (int j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```

2.38 Subiect 2021. Varianta TEST ANTRENAMENT 12

Scrieți un program C/C++ care citește de la tastatură numere naturale: n ($n \in [2, 20]$), k ($k \in [1, n]$) și $n \cdot n$ numere din intervalul $[0, 10^9]$, elemente ale unui tablou bidimensional cu n linii și n coloane. Programul transformă tabloul în memorie, deplasând circular spre dreapta, cu câte o poziție, toate elementele situate pe linia a k -a, în stânga diagonalei secundare, ca în exemplu. Elementele tabloului obținut sunt afișate pe ecran, fiecare linie pe câte o linie a ecranului, cu elementele fiecărei linii separate prin câte un spațiu.

Exemplu: pentru $n=5$, $k=2$ și tabloul

2	3	4	5	6
2	4	6	8	0
7	8	9	0	1
3	5	7	9	1
7	3	8	5	6

se obține tabloul

2	3	4	5	6
6	2	4	8	0
7	8	9	0	1
3	5	7	9	1
7	3	8	5	6

(10p.)

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

int main()
{
    int n, k, a[21][21];
    cin >> n >> k;
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
            cin >> a[i][j];
    int v = a[k-1][n-k-1];
    for (int j = n-k-1; j >= 1; j--)
        a[k-1][j] = a[k-1][j-1];
    a[k-1][0] = v;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < n; j++)
            cout << a[i][j] << ' ';
        cout << '\n';
    }
    return 0;
}
```

2.39 Subiect 2020. Varianta 5. Sub. II. 3

Variabilele i, j și k sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 6 linii și 4 coloane, numerotate începând de la 0, cu elemente numere naturale din intervalul $[0, 50]$.

Scrieți o secvență de instrucțiuni în urma executării căreia să se rearanjeze elementele pare situate pe coloana a 3-a a tabloului, astfel încât acestea să fie ordonate crescător, de sus în jos, în cadrul aceleiași coloane, iar celelalte elemente ale tabloului să își păstreze pozițiile, ca în exemplu.

Exemplu: pentru tabloul

2	3	4	5
8	7	8	5
1	3	5	7
3	0	2	9
3	5	1	6
7	3	0	2

se obține tabloul

2	3	0	5
8	7	2	5
1	3	5	7
3	0	4	9
3	5	1	6
7	3	8	2

```
for(i=0; i<5; i++)
    if(a[i][2]%2==0)
        for(j=i+1; j<6; j++)
            if(a[j][2]%2==0 && a[i][2]>a[j][2])
                swap(a[i][2],a[j][2]);
```

2.40 Subiect 2020. Varianta 2

Variabilele i și j sunt de tip întreg, iar variabila m memorează un tablou bidimensional cu 7 linii și 7 coloane, numerotate de la 0 la 6, cu elemente de tip char. Inițial, fiecare element memorează caracterul *. Fără a utiliza alte variabile, scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila m să memoreze tabloul alăturat.

for(i=0; i<7; i++)

for(j=0; j<7; j++)

.....

(6p.)

a	b	c	d	e	f	g
+	a	b	c	d	e	f
+	+	a	b	c	d	e
+	+	+	a	b	c	d
+	+	+	+	a	b	c
+	+	+	+	+	a	b
+	+	+	+	+	+	a

```
for(i=0; i<7; i++)
    for(j=0; j<7; j++)
        if(j<i)
            m[i][j]='+';
        else
            m[i][j]='a'+j-i;
```


2.41 Subject 2020. Varianta 6

Variabilele `i` și `j` sunt de tip întreg, iar variabila `a` memorează un tablou bidimensional cu 9 linii și 9 coloane, numerotate de la 0 la 8, având inițial toate elementele egale cu simbolul egal (=).

Scrieți secvența de mai jos, înlocuind punctele de suspensie, astfel încât, în urma executării secvenței obținute, variabila a să memoreze tabloul alăturat.

```
for (i=0; i<9; i++)
```

```
for (j=0;j<9;j++)
```

• • • • •

(6p.)

A 10x10 grid of arrows pointing right. The bottom four rows contain a staircase pattern of gray blocks. Row 6 (from top) has 1 gray block at the start. Row 7 has 2 gray blocks. Row 8 has 3 gray blocks. Row 9 has 4 gray blocks. The remaining 6 rows (rows 1-5) have no gray blocks.

```
for (i=0; i<9; i++)
    for (j=0; j<9; j++)
        if (i>j && i+j>8)
            a[i][j]='<';
        else
            a[i][j]='>';
```


2.42 Subiect 2020. Varianta TEST ANTRENAMENT 1

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 20]$, n și k , și construiește în memorie un tablou bidimensional cu n linii și $n \cdot k$ coloane, numerotate începând cu 1, astfel încât fiecare linie i ($i \in [1, n]$) memorează un șir crescător de termeni cu proprietatea că primul termen este i , fiecare valoare apare în șir de exact k ori și oricare doi termeni alăturați au valori egale sau consecutive.

Programul afișează pe ecran tabloul construit, fiecare linie a tabloului pe câte o linie a ecranului, cu valorile aflate pe aceeași linie separate prin câte un spațiu.

Exemplu: dacă $n=4$ și $k=3$, se afișează pe ecran tabloul alăturat.

1	1	1	2	2	2	3	3	3	4	4	4
2	2	2	3	3	3	4	4	4	5	5	5
3	3	3	4	4	4	5	5	5	6	6	6
4	4	4	5	5	5	6	6	6	7	7	7

```
#include <iostream>
using namespace std;
int n, k, a[21][21];
int main()
{
    int i, j;
    cin >> n >> k;
    for (i = 1; i <= n; i++)
    {
        a[i][1] = i;
        for (j = 2; j <= n * k; j++)
        {
            a[i][j] = a[i][j - 1];
            if (j % k == 1)
                a[i][j] = a[i][j - 1] + 1;
        }
    }
    for (i = 1; i <= n; i++)
    {
        for (j = 1; j <= n * k; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
}
```

2.43 Subiect 2020. Varianta TEST ANTRENAMENT 2. Sub II.3

Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 5 linii și 7 coloane, numerotate începând de la 0, cu elemente numere întregi, inițial toate egale cu 1.

Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni prin care se transformă în memorie tabloul, astfel încât orice element aflat pe prima linie sau pe prima coloană să aibă valoarea 1 și oricare alt element din tablou să fie egal cu ultima cifră a sumei celor două elemente alăturate lui, aflate pe aceeași linie dar pe coloana din stânga, respectiv pe aceeași coloană, dar pe linia anterioară. (6p.)

1	1	1	1	1	1	1
1	2	3	4	5	6	7
1	3	6	0	5	1	8
1	4	0	0	5	6	4
1	5	5	5	0	6	0

```
for ( int i = 1; i < 5; i++)
    for ( int j = 1; j < 7; j++)
        a[i][j] = (a[i][j-1] + a[i-1][j]) % 10;
```

2.44 Subiect 2020. Varianta TEST ANTRENAMENT 3

Scrieți un program C/C++ care citește de la tastatură un număr natural, n ($n \in [2, 20]$), și construiește în memorie un tablou bidimensional cu n linii și n coloane, având proprietățile:

- toate elementele situate pe diagonala secundară sunt nule;
- fiecare linie conține, începând cu diagonala secundară, de la dreapta la stânga, un șir strict crescător de numere consecutive, iar începând cu diagonala secundară, de la stânga la dreapta, tot un șir strict crescător de numere consecutive.

Programul afișează pe ecran tabloul construit, fiecare linie a tabloului pe câte o linie a ecranului, cu elementele aflate pe aceeași linie separate prin câte un spațiu.

Exemplu: dacă $n=5$ se afișează pe ecran tabloul alăturat.

(10p.)

4	3	2	1	0
3	2	1	0	1
2	1	0	1	2
1	0	1	2	3
0	1	2	3	4

```
#include <iostream>
using namespace std;
int main()
{
    int n, a[21][21];
    cin >> n;
    for (int i = 1; i <= n; i++)
        for (int j = 1; j <= n+1-i; j++)
        {
            a[i][j] = n-i-j+1;
            a[n+1-j][n+1-i] = a[i][j];
        }
    for (int i = 1; i <= n; i++)
    {
        for (int j = 1; j <= n; j++)
            cout << a[i][j] << " ";
        cout << '\n';
    }
    return 0;
}
```

2.45 Subiect 2020. Varianta TEST ANTRENAMENT 4

Într-un tablou bidimensional, cu elemente având valori numai în mulțimea $\{0,1\}$, numim coloane „complementare” două coloane cu proprietatea că oricare două elemente ale acestora, aflate pe aceeași linie, sunt diferite.

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2,20]$, m și n , și elementele unui tablou bidimensional cu m linii și n coloane, numere naturale din mulțimea $\{0,1\}$. Programul afișează pe ecran numărul de coloane ale tabloului care sunt „complementare” cu prima coloană a acestuia.

Exemplu: dacă $m=3$, $n=6$, pentru tabloul alăturat se afișează pe ecran 3.

(10p.)

1	1	0	0	1	0
0	1	1	1	1	1
1	0	0	0	1	0

```
#include <iostream>
using namespace std;
int n, a[21][21], m, ok;
int main()
{
    int i, j, nr=0;
    cin >> m >> n;
    for (i=1; i<=m; i++)
        for (j=1; j<=n; j++)
            cin >> a[i][j];
    for (j=2; j<=n; j++)
    {
        ok=1;
        for (i=1; i<=m; i++)
            if (a[i][j]==a[i][1])
                ok=0;
        if (ok==1)
            nr++;
    }
    cout << nr;
}
```

2.46 Subiect 2020. Varianta TEST ANTRENAMENT 5. Sub. II.3

Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 6 linii și 6 coloane, numerotate de la 0 la 5, având inițial toate elementele egale cu caracterul @. Fără a utiliza alte variabile, scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila a să memoreze tabloul alăturat.

```
for (i=0; i<6; i++)
    for (j=0; j<6; j++)
```

.....

(	(	(	)	)	)
*	(	(	)	)	*
*	*	(	)	*	*
*	*	(	)	*	*
*	(	(	)	)	*
(	(	(	)	)	)

(6p.)

```
for (i=0; i<6; i++)
    for (j=0; j<6; j++)
        if (i==0 || i==5 || j==2 || j==3 ||
            ((i==1 || i==4) && (j==1 || j==4)))
            if (j<3)
                a[i][j]='(';
            else
                a[i][j]=')';
        else
            a[i][j]='*';
```

2.47 Subiect 2020. Varianta TEST ANTRENAMENT 6. Sub. II.3

Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 4 linii și 5 coloane, numerotate începând de la 0, cu elemente numere întregi, inițial toate nule. Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni astfel încât, în urma executării acestora, variabila a să memoreze tabloul alăturat.

16	17	18	19	20
11	12	13	14	15
6	7	8	9	10
1	2	3	4	5

(6p.)

```
for (i = 0; i < 4; i++)
    for (j = 0; j < 5; j++)
        a[i][j] = (3 - i) * 5 + j + 1;
```

2.48 Subiect 2020. Varianta TEST ANTRENAMENT 7

O valoare k **polarizează** două şiruri dacă există doi termeni care au acea valoare, unul fiind în primul şir, iar celălalt în al doilea şir.

Scrieţi un program C/C++ care citeşte de la tastatură numere naturale din intervalul $[1, 20]$: m , n şi elementele unui tablou bidimensional cu m linii şi n coloane, cu proprietatea că nu există două elemente egale situate pe aceeaşi linie sau pe aceeaşi coloană. Programul afişează pe ecran valorile care pot polariza două şiruri, şi anume şirul format din elementele de pe prima coloană, respectiv şirul format din elementele ultimei coloane a tabloului.

Valorile sunt afişate într-o ordine oarecare, separate prin câte un spaţiu, iar dacă nu există astfel de valori se afişează pe ecran mesajul **nepolarizate**.

Exemplu: pentru $m=4$, $n=5$ şi tabloul alăturat se afişează pe ecran, nu neapărat în această ordine, numerele 5 6

3	7	1	2	<u>5</u>
2	4	5	9	<u>6</u>
<u>6</u>	2	7	8	1
<u>5</u>	3	2	7	8

(10p.)

```
#include <iostream>
using namespace std;
int n,m,a[21][21];
int main()
{
    int i,j,x,y,nr=0;
    cin>>m>>n;
    for(i=1; i<=m; i++)
        for(j=1; j<=n; j++)
            cin>>a[i][j];
    ///numar cate elem polarizeaza pe prima si ult col
    for(i=1; i<=m; i++) ///ma uit pe prima col
        ///elem de pe linia i din prima col il caut pe col n
        for(j=1; j<=m; j++)
            if(a[i][1]==a[j][n])
            {
                if(nr==0)
                    x=a[i][1];
                else
                    if(nr==1)
                        y=a[i][1];
                nr++;
                break;
            }
    if(nr==2)
        cout<<x<<" "<<y;
    else
        cout<<"nepolarizate";
}
```


2.49 Subiect 2020. Varianta TEST ANTRENAMENT 8

Scrieți un program C/C++ care citește de la tastatură numere naturale: n ($n \in [2, 20]$), k ($k \in [1, n]$) și $n \cdot n$ numere din intervalul $[0, 10^9]$, elemente ale unui tablou bidimensional cu n linii și n coloane, numerotate începând de la 1. Programul transformă tabloul în memorie, deplasând circular spre stânga, cu câte o poziție, toate elementele situate pe linia k , în stânga diagonalei principale, ca în exemplu. Elementele tabloului obținut sunt afișate pe ecran, fiecare linie pe câte o linie a ecranului, cu elementele fiecărei linii separate prin câte un spațiu.

Exemplu: pentru $n=6$, $k=4$ și tabloul

1	2	3	4	5	6
6	7	8	9	0	1
2	4	6	8	0	2
1	3	5	7	9	5
6	2	7	4	5	7
3	9	4	8	5	7

se obține tabloul

1	2	3	4	5	6
6	7	8	9	0	1
2	4	6	8	0	2
3	5	1	7	9	5
6	2	7	4	5	7
3	9	4	8	5	7

```
#include <iostream>
using namespace std;
int n, k, i, j, a[21][21];
int main()
{
    int i, j;
    cin >> n >> k;
    for (i = 1; i <= n; i++)
        for (j = 1; j <= n; j++)
            cin >> a[i][j];
    int v = a[k][1];
    for (j = 2; j <= k - 1; j++)
        a[k][j-1] = a[k][j];
    a[k][k-1] = v;
    for (i = 1; i <= n; i++)
    {
        for (j = 1; j <= n; j++)
            cout << a[i][j] << ' ';
        cout << '\n';
    }
    return 0;
}
```

2.50 Subiect 2020. Varianta TEST ANTRENAMENT 10. Sub II.3

Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 4 linii și 5 coloane, numerotate începând de la 0, cu elemente numere întregi, inițial toate nule. Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni astfel încât, în urma executării acesteia, variabila a să memoreze tabloul alăturat. (6p.)

5	4	3	2	1
10	9	8	7	6
15	14	13	12	11
20	19	18	17	16

```
for (i = 0; i < 4; i++)
    for (j = 0; j < 5; j++)
        a[i][j] = (i + 1) * 5 - j;
```

2.51 Subiect 2020. Varianta TEST ANTRENAMENT 11

Variabilele i și p sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 7 linii și 7 coloane, cu elemente numere întregi din intervalul $[0, 10^2]$.

Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni astfel încât, în urma executării acesteia, să se afișeze, separate prin câte un spațiu, produsul celor 7 elemente situate pe diagonala principală a tabloului, urmat de produsul elementelor situate simultan pe diagonala principală și pe primele 6 linii ale tabloului, și așa mai departe, astfel încât ultimul număr afișat să fie elementul situat simultan pe diagonala principală și pe prima linie a tabloului.

Exemplu: pentru tabloul alăturat se afișează 1680 840 420 84 21 7 1

1	8	3	9	6	5	5
0	7	4	2	5	5	4
8	6	3	5	1	2	3
2	4	9	4	8	3	4
2	1	7	5	5	5	6
7	4	2	4	9	2	7
0	9	6	3	1	7	2

(6p.)

```
p = 1;
for (i = 0; i < 7; i++)
    p = p * a[i][i];
for (i = 6; i >= 0; i--)
{
    cout << p << " ";
    p = p / a[i][i];
}
```


2.52 Subiect 2020. Varianta TEST ANTRENAMENT 13

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 20]$, n și k , și construiește în memorie un tablou bidimensional cu $n \cdot k$ linii și n coloane, numerotate începând cu 1, astfel încât fiecare coloană i ($i \in [1, n]$) memorează un șir crescător de termeni cu proprietatea că primul termen este i , fiecare valoare apare în șir de exact k ori și oricare doi termeni alăturați au valori egale sau consecutive.

Programul afișează pe ecran tabloul construit, fiecare linie a tabloului pe câte o linie a ecranului, cu valorile aflate pe aceeași linie separate prin câte un spațiu.

Exemplu: dacă $n=4$ și $k=3$, se afișează pe ecran tabloul alăturat.

(10p.)

1	2	3	4
1	2	3	4
1	2	3	4
2	3	4	5
2	3	4	5
2	3	4	5
3	4	5	6
3	4	5	6
3	4	5	6
4	5	6	7
4	5	6	7
4	5	6	7

```
#include <iostream>
using namespace std;
int a[401][21], n, k;
int main()
{
    int i, j, nr, p=0;
    cin >> n >> k;
    nr = -1;
    do
    {
        nr++;
        for(i=1; i<=k; i++)
        {
            p++;
            for(j=1; j<=n; j++)

                a[p][j] = nr+j;
        }
    }
    while (nr<n);
    for(i=1; i<=n*k; i++)
    {
        for(j=1; j<=n; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
}
```

2.53 Subiect 2020. Varianta TEST ANTRENAMENT 14

Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 5 linii și 7 coloane, numerotate începând de la 0, cu elemente numere întregi, inițial toate egale cu 0.

Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni prin care se transformă în memorie tabloul, astfel încât orice element aflat pe prima linie sau pe ultima coloană să aibă valoarea 1 și oricare alt element din tablou să fie egal cu ultima cifră a sumei celor două elemente alăturate lui, aflate pe aceeași linie dar pe coloana din dreapta, respectiv pe aceeași coloană, dar pe linia anterioară. (6p.)

1	1	1	1	1	1	1
7	6	5	4	3	2	1
8	1	5	0	6	3	1
4	6	5	0	0	4	1
0	6	0	5	5	5	1

```
for (i = 0; i < 5; i++)
    for (j = 6; j >= 0; j--)
        if (i == 0 || j == 6)
            a[i][j] = 1;
        else
            a[i][j] = (a[i][j + 1] + a[i - 1][j]) % 10;
```

2.54 Subiect 2020. Varianta TEST ANTRENAMENT 15

Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 4 linii și 5 coloane, numerotate începând de la 0, cu elemente numere întregi, inițial toate nule. Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni astfel încât, în urma executării acestora, variabila a să memoreze tabloul alăturat. (6p.)

1	5	9	13	17
2	6	10	14	18
3	7	11	15	19
4	8	12	16	20

```
for (i = 0; i < 4; i++)
    for (j = 0; j < 5; j++)
        a[i][j] = j * 4 + i + 1;
```

2.55 Subiect 2020. Varianta TEST ANTRENAMENT 16

Scrieți un program C/C++ care citește de la tastatură două numere naturale din intervalul $[2, 10^2]$, n și m , și construiește în memorie un tablou bidimensional cu n linii și m coloane, cu proprietatea că parcurgându-l linie cu linie de sus în jos și fiecare linie de la stânga la dreapta, se obține șirul primelor $n \cdot m$ pătrate perfecte impare, ordonat strict descrescător, ca în exemplu.

Elementele tabloului obținut se afișează pe ecran, fiecare linie a tabloului pe câte o linie a ecranului, valorile de pe aceeași linie fiind separate prin câte un spațiu.

Exemplu: pentru $n=2$, $m=3$ se obține tabloul alăturat.

(10p.)

121	81	49
25	9	1

```
#include <iostream>
using namespace std;
int a[103][103], n, m;
int main()
{
    cin >> n >> m;
    int i, j, x = -1;
    for (i = n; i >= 1; i--)
        for (j = m; j >= 1; j--)
        {
            x = x+2;
            a[i][j] = x*x;
        }
    for (i = 1; i <= n; i++)
    {
        for (j = 1; j <= m; j++)
            cout << a[i][j] << " ";
        cout << endl;
    }
}
```

2.56 Subiect 2020. Varianta TEST ANTRENAMENT 17

Variabilele i și j sunt de tip întreg, iar variabila a memorează un tablou bidimensional cu 4 linii și 5 coloane, numerotate începând de la 0, cu elemente numere întregi, inițial toate nule. Fără a utiliza alte variabile decât cele menționate, scrieți o secvență de instrucțiuni C/C++ astfel încât, în urma executării acesteia, variabila a să memoreze tabloul alăturat.

20	16	12	8	4
19	15	11	7	3
18	14	10	6	2
17	13	9	5	1

(6p.)

```
for (i = 0; i < 4; i++)
    for (j = 0; j < 5; j++)
        a[i][j] = 20 - 4 * j - i;
```

2.57 Subiect 2020. Varianta TEST ANTRENAMENT 19

Într-un tablou bidimensional, cu elemente având valori numai în mulțimea $\{0, 1\}$, numim linii „complementare” două linii cu proprietatea că oricare două elemente ale acestora, aflate pe aceeași coloană, sunt diferite.

Scrieți un program C/C++ care citește de la tastatură trei numere naturale m , n și k ($m, n \in [2, 20]$, $k \in [1, m]$) și elementele unui tablou bidimensional cu m linii și n coloane, numere naturale din mulțimea $\{0, 1\}$. Programul afișează pe ecran mesajul **DA** dacă există cel puțin o linie „complementară” cu linia a k -a a acestuia, sau mesajul **NU** în caz contrar.

Exemplu: pentru $m=7$, $n=3$, $k=2$ și tabloul alăturat se afișează pe ecran mesajul **DA**. (10p.)

0	1	0
1	1	0
0	0	0
0	0	1
0	0	1
1	1	1
0	0	1

```
#include <iostream>
using namespace std;
int main()
{
    int m, n, k, i, j, a[21][21], ok, gasit = 0;
    cin >> m >> n >> k;
    for (i = 1; i <= m; i++)
        for (j = 1; j <= n; j++)
            cin >> a[i][j];
    for (i = 1; i <= m; i++)
    {
        if (i != k)
        {
            ok = 1;
            for (j = 1; j <= n; j++)
                if (a[i][j] == a[k][j])
                    ok = 0;
            if (ok == 1)
                gasit = 1;
        }
    }
    if (gasit)
        cout << "DA";
    else
        cout << "NU";
    return 0;
}
```


2.58 Subiect 2020. Varianta TEST ANTRENAMENT 20

Variabilele `i` și `j` sunt de tip întreg, iar variabila `a` memorează un tablou bidimensional cu 7 linii și 7 coloane, numerotate de la 0 la 6, având inițial toate elementele egale cu caracterul `*`. Fără a utiliza alte variabile, scrieți secvența de instrucțiuni de mai jos, înlocuind punctele de suspensie astfel încât, în urma executării secvenței obținute, variabila `a` să memoreze tabloul alăturat.

```
for(i=0;i<7;i++)
    for(j=0;j<7;j++)
        .....
```

b	a	a	a	a	a	b
b	b	a	a	a	b	b
b	b	b	a	b	b	b
b	b	b	b	b	b	b
b	b	b	a	b	b	b
b	b	a	a	a	b	b
b	a	a	a	a	a	b

(6p.)

```
for(i=0; i<7; i++)
    for(j=0; j<7; j++)
        if ((j>i && j<6-i) || (j<i && j>6-i))
            a[i][j]='a';
        else
            a[i][j]='b';
```