Universitatea Babeș-Bolyai, Facultatea de Matematică și Informatică

Consultații la Informatică pentru pregătirea concursului de admitere 2021

12 decembrie 2020

Asist. univ. drd. Florentin Bota

Tablouri bidimensionale (matrici)

1. Rezolvați cerința de mai jos:
   - Declarați o matrice cu valorile:

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

   - Afișați elementele din matrice de sub diagonala principală

Soluție C++

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

// Dimensiune matrice (conform cerintei)
const int N = 4;

// Afisare matrice, optionala
void afisareMatrice(int a[N][N]){

    for (int i = 0; i < N; i++){
        for(int j = 0; j < N; j++)
            cout << a[i][j]<<" ";
        cout << "\n";
    }
}


// Afisarea elementelor din matrice de sub diagonala principala
void afisareElemente(int a[N][N]){

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

```
    }

}

// Functia principala
int main(){

    // Declarare matrice
    int a[N][N]= {
        {1,  2,  3,  4 },
        {5,  6,  7,  8 },
        {9,  10, 11, 12},
        {13, 14, 15, 16}
    };

    afisareMatrice(a);
    afisareElemente(a);


    system("pause");
    return 0;
}
```

Soluție Pascal

```
program matriceEx1;

var
    a: array[1..4, 1..4] of integer = ( (1,2,3,4),
                                        (5,6,7,8),
                                        (9,10,11,12),
                                        (13,14,15,16));
    i,j: integer;

begin
    writeln('Hello there');

    for i:=1 to 4 do
    begin
        for j:=1 to i-1 do
            write(a[i][j], ' ');
        writeln;
    end;

    readln;
end.
```

2. Citiți de la tastatură un număr n <= 6.

   - Generați o matrice de forma `A[n][n]` care să conțină elementele din șirul lui Fibonacci. Ex:

> n = 4

1	1	2	3
5	8	13	21
34	55	89	144
233	377	610	987

Solutie C++

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

// Generam matricea conform cerintei
// Obs: Solutie implementata in timpul consultatiilor
void genereazaMatrice(unsigned int a[7][7], unsigned short n){

    // Initializam primele 2 elemente
    a[1][1] = a[1][2] = 1;

    // Parcurgem liniile
    for(int i=1; i<=n; i++)

        // Prima linie, putem incepe de la j = 3
        if(i==1)
            for(int j=3; j<=n; j++)
                a[i][j] = a[i][j-2] + a[i][j-1];
        else
        {   // Daca suntem pe prima coloana
            for(int j=1; j<=n; j++)
                if(j==1)
                    a[i][j] = a[i-1][n] + a[i-1][n-1];
                else
                {   // Daca suntem pe coloana 2
                    if(j==2)
                        a[i][j] = a[i][j-1] + a[i-1][n];
                    // Daca suntem pe coloana 3 sau mai mare de 3
                    else
                    {
                        a[i][j] = a[i][j-2] + a[i][j-1];
                    }

                }

        }
```

```
}

// Varianta simplificata
void genereazaMatriceV2(unsigned int a[7][7], unsigned short n){
    unsigned int x, y;

    // Initializam
    x=y=0;

    // Generam valorile pe masura ce parcurgem matricea
    for(int i=1; i<=n; i++)
        for(int j=1; j<=n; j++){
            if(i==1 && j==1)
                a[i][j] = 1;
            else
                a[i][j] = x+y;
            x = y;
            y = a[i][j];
        }

}

// Afisare
void afisareMatrice(unsigned int a[7][7], unsigned short n){

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

int main(){

    unsigned short n;
    unsigned int a[7][7];

    cout<<"Dati n= ";
    cin>>n;

    //genereazaMatrice(a,n);
    genereazaMatriceV2(a,n);
    afisareMatrice(a,n);

    system("pause");
    return 0;
}
```

- Generați o matrice de forma `B[n][n]` care să respecte modelul de mai jos:

> n = 4

1	1	2	3
1	2	3	5
2	3	5	8
3	5	8	13

Solutie C++

```
#include <iostream>

using namespace std;

// Initializare
void generareMatrice(int a[7][7], int n){

    // Initializam primele 2 elemente din matrice
    a[1][1] = a[1][2] =1;

    // Prima linie
    for(int j=3; j<=n; j++)
        a[1][j] = a[1][j-1] + a[1][j-2];

    // Generare
    for(int i=2; i<=n; i++)
        for(int j=1; j<=n; j++)
            if(j==1)
                a[i][j] = a[i-1][j+1];
            else
                a[i][j] = a[i-1][j-1] + a[i-1][j];
}

// Afisare
void afisareMatrice(int a[7][7], int n){

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

int main()
{
    int a[7][7], n;

    // Citim n
    cout << "Dati n= ";
    cin>>n;
```

```
        // Generare
        generareMatrice(a,n);
        afisareMatrice(a,n);

        system("pause");
        return 0;
    }
```

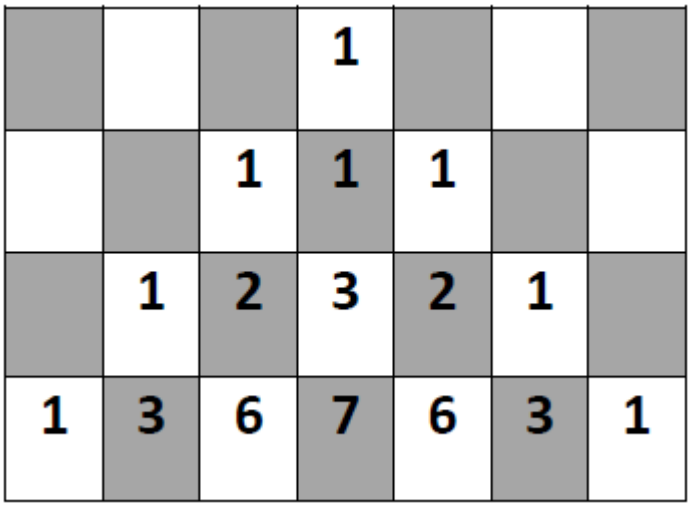
---

3. Sah

> ○ Enunț adaptat, OJI 2015, preluat din Consultații la Informatică, 7 decembrie 2019

Se considera o tabla de sah cu n+1 linii si 2n+1 coloane. Pe prima linie patratul din mijloc contine 1 gram de fan, iar celelalte patrate de pe prima linie nu contin nimic. Incepand cu linia a doua fiecare patrat contine o cantitate de fan obtinuta prin adunarea cantitatilor de fan din cele 3 patrate ale liniei anterioare cu care se invecineaza (pe verticala si diagonala).

De exemplu, daca n=3 tabla are 4 linii, 7 coloane si urmatoarea configuratie.



Un cal pleaca de pe prima linie, de pe o coloana k<=n, sare din orice pozitie (i,j) in pozitia (i+1,j+2) atat timp cat este posibil si mananca tot fanul din patratele prin care trece. De exemplu, pentru n=3 și k=1, patratele prin care trece calul sunt (0, 1), (1, 3) si (2, 5) iar cantitatea totala de fan este 0 + 1 + 1 = 2.

Cerinte

1. Determinati continutul matricii daca se citeste valoarea lui n.
2. Calculati cantitatea totala de fan de pe linia k a tablei de sah (se cunosc valorile lui n si k).
3. Calculati cate grame de fan mananca un cal care pleaca de pe prima linie, coloana k (se cunosc valorile lui n si k). Date de intrare

Se citesc valorile n si k, 0<= n <=100, 0 <= k <= 200

Date de ieşire

- Matricea de dimensiuni n+1, 2n+1
- Cantitatea totala de fan de pe linia k

- Cantitatea de fan consumata de un cal care pleaca din linia 0, coloana k

Exemplu

Date de intrare	Date de iesire
n = 3	Matricea: 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 1 2 3 2 1 0 1 3 6 7 6 3 1
k = 3	Cantitate linia k : 27
k = 0	Cantitate consumata de cal: 4

Pasii algoritmului principal

Algoritm tablaSah

- @ citeste un numar n
- @ genereaza matricea
- @ tipareste matricea
- @ citeste un numar k
- @ calculeaza cantitatea de pe linia k
- @ tipareste cantitatea
- @ citeste un numar k
- @ determina deplasarea calului si calculeaza cantitatea consumata
- @ tipareste cantitatea

Sf.Algoritm

Identificarea subalgoritmilor

- program principal
  - initializareMatrice
  - afisareMatrice
  - populareMatrice
  - cantitateLinie
  - traversareCal

Solutie C++

```
// Rezolvarea nu este optimizata pentru viteza de executie
// Rezolvarea exemplifica o abordare a problemei bazata pe descompunerea in
subprobleme
```

```
// Programul a fost compilat cu Visual Studio Code
#include <iostream>
using namespace std;

typedef struct {
    int n;
    int elem[101][201];
} Matrice;

// Initial, toate elementele sunt 0
void initializareMatrice(Matrice &a) {
    for (int i = 0; i < a.n + 1; i++) {
        for (int j = 0; j < 2 * a.n + 1; j++) {
            a.elem[i][j] = 0;
        }
    }
}

// Tiparire matrice pe ecran
void afisareMatrice(Matrice &a) {
    for (int i = 0; i < a.n + 1; i++) {
        for (int j = 0; j < 2*a.n + 1; j++)
            cout << a.elem[i][j] << " ";

        cout << endl;
    }

    cout << endl;
}


// Populatea matricii cu valori conform regulii date
void populareMatrice(Matrice &a) {

    a.elem[0][a.n] = 1;

    for (int i = 1; i < a.n + 1; i++) {

        // primul si ultimul element de pe linie trebuie tratate separat
        a.elem[i][0] = a.elem[i - 1][0] + a.elem[i - 1][1];
        a.elem[i][2 * a.n] = a.elem[i - 1][2 * a.n - 1] + a.elem[i - 1][2 * a.n];

        // calcul valoare elemente pe baza insumarii a 3 elemente de pe linia
precedenta
        for (int j = 1; j < 2 * a.n; j++) {
            a.elem[i][j] = a.elem[i - 1][j - 1] + a.elem[i - 1][j] + a.elem[i - 1]
[j + 1];
        }
    }
}
// Calcul suma elementelor de pe linia k pentru matricea populata
int cantitateLinie_v1(Matrice &a, int k) {
    int cantitate = 0;
```

```
    for (int j = 0; j < 2 * a.n + 1; j++) {
        cantitate += a.elem[k][j];
    }

    return cantitate;
}

// Returneaza 3^k
// Aceasta functie este necesara pentru a nu folosi functii din biblioteci
suplimentare (lucru // cerut in conditii de examen); inlocuieste asadar apelul
functiei pow(3, k).
int putere(int k)
{
    int p = 1;

    for (int i = 0; i < k; i++)
        p = p * 3;

    return p;
}

// Calcul suma elementelor tinand cont de faptul ca suma de pe o linie este tripla
fata de suma liniei anterioare.
// Suma primei linii este 1, deducem ca suma liniei k este 3*3*...*3 (de k ori),
unde k = 0 pentru prima linie.
int cantitateLinie_v2(Matrice &a, int k) {
    return putere(k);
}

// Calcul cantitate totala acumulata prin traversarea matricii pornind din linia
0, coloana k
// conform regulii calului pe o tabla de sah
int traversareCal(Matrice &a, int k) {
    int suma = 0;
    int i = 0;
    int j = k;

    while (i < a.n + 1 && j < 2 * a.n + 1) {
        suma += a.elem[i][j];
        i = i + 1;
        j = j + 2;
    }

    return suma;
}

int  main() {

    Matrice a;

    cout << "n = ";
    cin >> a.n;

    initializareMatrice(a);
```

```
    populareMatrice(a);
    afisareMatrice(a);

    int k = -1;

    cout << "Introduceti k = ";
    cin >> k;

    cout << "Cantitate linia " << k << " este (v1) " << cantitateLinie_v1(a, k) <<
endl;
    cout << "Cantitate linia " << k << " este (v2) " << cantitateLinie_v2(a, k) <<
endl;

    cout << "Introduceti coloana initiala cal = ";
    cin >> k;

    cout << "Cantitatea adunata de cal este " << traversareCal(a, k) << "\n";

    system("pause");

    return 0;
}
```
