Ministerul Educaţiei și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică şi Microelectronică

**RAPORT**

Lucrare de laborator Nr.4

*la Programarea Calculatoarelor*

Tema: Prelucrarea tablourilor bidimensionale (matricelor) în limbajul C

A efectuat: st. gr. SI-212 Șeremet Alexandru

A verificat: lect. asist. Mititelu Vitalii

Chişinău 2021

**Lucrare de laborator Nr.4**

**Tema:** Prelucrarea tablourilor bidimensionale (matricelor) în limbajul C

**Scopul:** Studierea posibilităților și mijloacelor limbajului C pentru programarea algoritmilor de prelucrare a tablourilor bidimensionale.

**Sarcina:** Scrieți un program care citește de la tastatură numărul n>1 de rânduri și numărul m>1 de coloane ale tabloului bidimensional (mnatricei), apoi citește de la tastatură aceste n X m elemente ale tabloului, efectuează calculele indicate în variantă și afișează pe ecran rezultatul:

**Varianta 15:**

Să se efectueze uniformizarea tabloului. Operația de uniformizare a tabloului se efectuează prin înlocuirea fiecărui element cu media aritmetică a tuturor vecinilor.

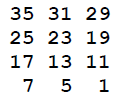
**Rezumat:**

# Tablouri bidimensionale – Matrici

O matrice este o forma de organziare a datelor de acelasi tip. O matrice reprezinta un tablou bidimensional in care sunt stocate date de acelasi tip. Elementele dintr-o matrice pot fi identificate dupa linia si coloana pe care se afla.

Se numeste matrice cu m linii si n coloane, un tablou cu m linii si n coloane.

Pentru m=4 si n=3 se obtine o matrice cu 4 linii si 3 coloane, spre exemplu:



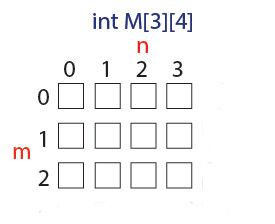
In exemplul dat, numarul 35 se afla pe linia 1, coloana 1, pe cand 13 se afla pe linia 3, coloana 2. Am considerat numerotarea liniilor si coloanelor incepand de la 1. Daca matricea se va citi, luand doi contori ( i si j) cu valori de la 0 la m – 1 (pentru m linii), respectiv de la 0 la n – 1 (pentrul n coloane), atunci 35 se va afla pe linia 0 si coloana 0.

In limbajul C++ declararea unei matrici se realizeaza prin specificarea tipului elementelor din matricii, indentificatorul matricii, urmat apoi de dimensiunea acesteia (numarul de linii si de coloane) intre paranteze drepte.

Spre exemplu:

* int A[4][3]; (declara o matrice de 4 linii si 3 coloane continand numere intregi)
* float B[5][10]; (declara o matrice de 5 linii si 10 coloane continand numere reale)
* char C[20][12]; (declara o matrice de 20 de linii si 12 coloane continand caractere alfanumerice)

Observatie: Numerotarea liniilor si coloanelor unei matrici incepe de la valoarea 0. Astfel, prima linie si prima coloana a matricii este linia si coloana 0, a doua coloana este 1, etc.



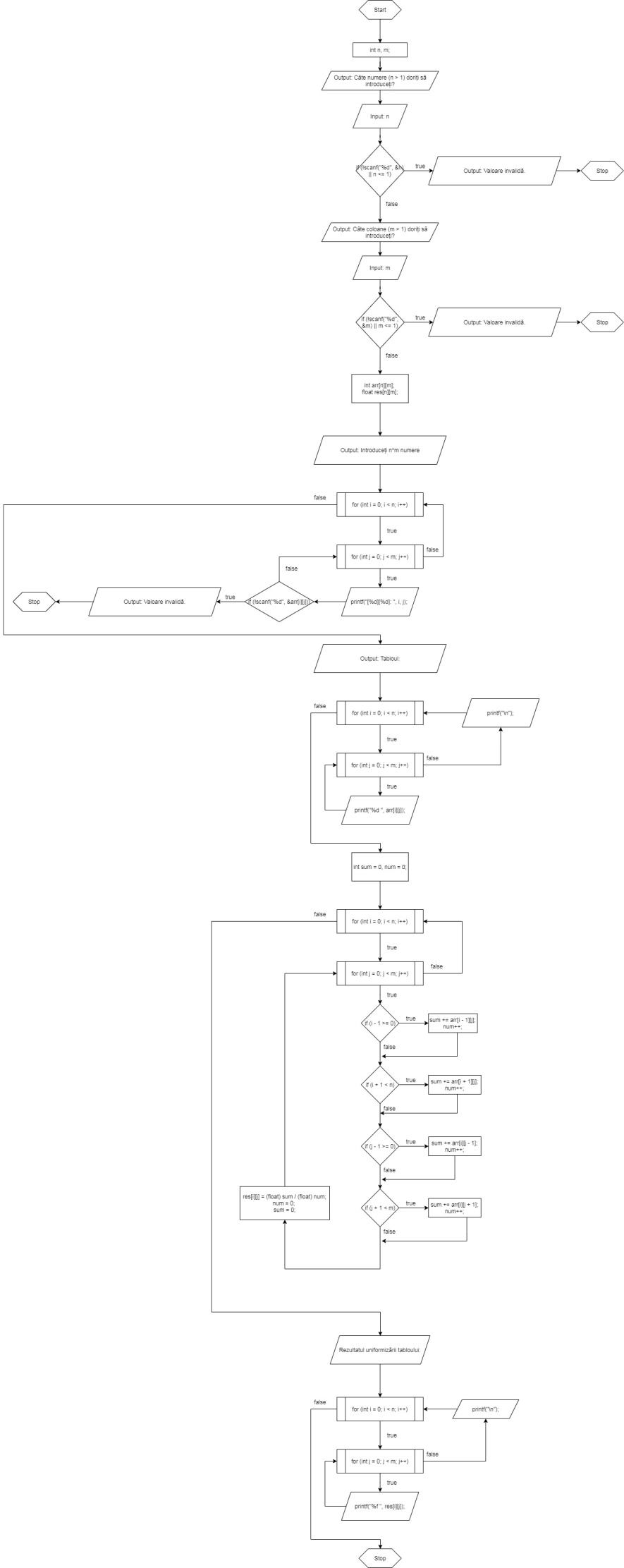
Observatie: Putem numerota liniile si coloanele de la 1, in acest caz, pentru a nu avea probleme de memorie, declaram matricea cu un numar in plus de linii si coloane. Astfel, daca vrem sa salvam o matrice cu 3 linii si 4 coloane, incepand numerotarea de la 1, vom scrie m[4][5] in loc de m[3][4].

Mai jos putem vedea citirea si afisare unei matrici salvata cu numerotoarea incepand de la 1.

|  |  |
| --- | --- |
|  | #include <iostream> |
|  | using namespace std; |
|  |  |
|  | int a[10][20],n,m,i,j; |
|  |  |
|  | int main() |
|  | { |
|  | cin>>m>>n; //se citesc numarul de linii si coloane |
|  | //citirea pe rand a elementelor matricii |
|  | for(i=1;i<=m;i++) |
|  | for(j=1;j<=n;j++) |
|  | cin>>a[i][j]; |
|  | //afisarea matricii |
|  | for(i=1;i<=m;i++) |
|  | {for(j=1;j<=n;j++) |
|  | cout<<a[i][j]<<" "; |
|  | cout<<endl; |
|  | } |
|  | } |

Matricile au un caz particular, atunci cand numarul de linii este egal cu numarul de coloane, acele matrici numindu-se matrici patratice.

**Schema bloc a algoritmului de soluționare a problemei:**



**Codul deplin al programului:**

#include <stdio.h>

int main () {

    int n, m;

    printf ("\nCâte rânduri (n > 1) doriți să introduceți? ");

    if (!scanf("%d", &n) || n <= 1) {

        printf ("Valoare n invalidă.\n");

        return (1);

    }

    printf ("\nCâte coloane (m > 1) doriți să introduceți? ");

    if (!scanf("%d", &m) || m <= 1) {

        printf ("Valoare m invalidă.\n");

        return (1);

    }

    int arr[n][m];

    float res[n][m];

    printf("\nIntroduceți %d numere: \n", n\*m);

    for (int i = 0; i < n; i++){

        for (int j = 0; j < m; j++) {

            printf("[%d][%d]: ", i, j);

            if (!scanf("%d", &arr[i][j])) {

                printf ("Valoare invalidă.\n");

                return (1);

            }

        }

    }

    printf("\n");

    printf("Tabloul:\n");

    for (int i = 0; i < n; i++){

        for (int j = 0; j < m; j++) {

            printf("%d ", arr[i][j]);

        }

        printf("\n");

    }

    printf("\n");

    //printf("Calcule:\n");

    int sum = 0, num = 0;

    for (int i = 0; i < n; i++){

        for (int j = 0; j < m; j++) {

            if (i - 1 >= 0) {

                sum += arr[i - 1][j];

                num++;

                //printf(" %d ", arr[i - 1][j]);

            }

            if (i + 1 < n) {

                sum += arr[i + 1][j];

                num++;

                //printf(" %d ", arr[i + 1][j]);

            }

            if (j - 1 >= 0) {

                sum += arr[i][j - 1];

                num++;

                //printf(" %d ", arr[i][j - 1]);

            }

            if (j + 1 < m) {

                sum += arr[i][j + 1];

                num++;

                //printf(" %d ", arr[i][j + 1]);

            }

            //printf(" => %d / %d", sum, num);

            res[i][j] = (float) sum / (float) num;

            //printf(" = %f;\n", res[i][j]);

            num = 0;

            sum = 0;

        }

    }

    //printf("\n");

    printf("Rezultatul uniformizării tabloului:\n");

    for (int i = 0; i < n; i++){

        for (int j = 0; j < m; j++) {

            printf("%f ", res[i][j]);

        }

        printf("\n");

    }

    return (0);

}

Liniile transformate în comentarii reprezintă codul care afișează calculele efectuate.

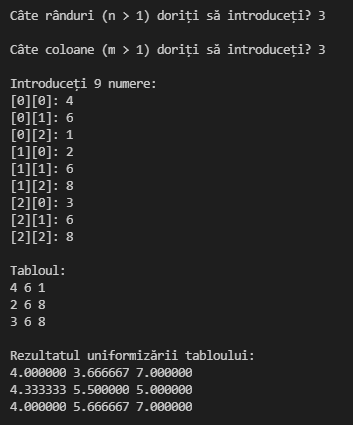
**Execuția programului:**

* Set de date nr. 1: n = 3, m = 3,

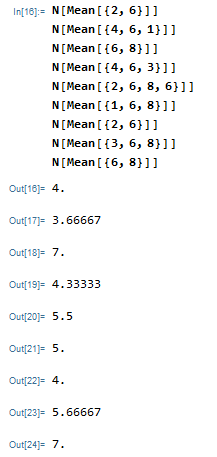
arr[n][m] = {4 6 1

2 6 8

3 6 8}



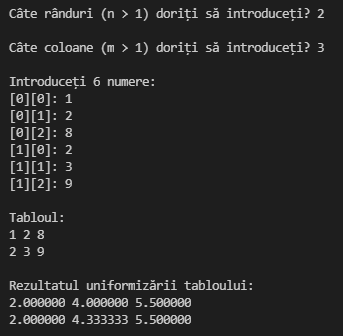
Calcule confirmative în Wolfram:



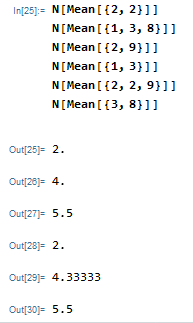
* Set de date nr. 2: n = 2, m = 3,

arr[n][m] = {1 2 8

2 3 9}



Calcule confirmative în Wolfram:



* Set de date nr. 3: n = „Nu”

C:\Users\Sand\Desktop\Capture.PNG

„Nu” este de tip char \*, ceea ce nu este int și dă eroare ca rezultat. Același lucru va avea loc și dacă n sau m <= 1, sau dacă în orice etapă este introdus orice alt tip de date decât int.

**Concluzii:**

1. Tablourile bidimensionale sunt, în fond, tablouri unidimensionale formate din tablouri unidimensonale. Într-un mod similar se pot crea tablouri tridimensionale sau chiar de orice altă dimensiune, adaugând încă un « [] » în linia ce îl declară.
2. Pentru a manipula efectiv tablourile, este nevoie de cunoștințe despre instrucțiunile ciclice, deoarece, foarte des, operațiile cu tablourile necesită acțiuni repetitive aplicate asupra fiecărui element, sau asupra unui număr selectiv de elemente.
3. Importanța instrucțiunilor ciclice este și mai mare în cazul tablourilor bidimensionale, deoarece apare necesitatea de a opera cu tablouri din alte tablouri într-un mod uniformizat.

**Surse:**

* **invata.info, curs la C++.** Capitolul: Tablouri bidimensionale – Matrici

https://invata.info/2017/02/17/tablouri-bidimensionale-matrici/