

8. Aplicații cu tablouri în C/C++. (C/C++ applications using arrays)

1. Obiective:

- Înțelegerea tablourilor unidimensionale și multidimensionale din C/C++
- Înțelegerea modului în care se lucrează cu tablourilor unidimensionale și multidimensionale din C/C++: declarare, inițializare, accesul la elemente, diferite operații
- Scrierea și rularea de programe simple în care sunt folosite tablouri

1'. Objectives:

- Understanding the unidimensional and multidimensional arrays in C/C++
- Understanding the arrays' operating mode: declaration, initialization, elements' access, various operations
- Writing and running some simple programs that use arrays

2. Breviar teoretic

Un tablou este o listă de elemente de același tip stocate contigu. Dimensiunea tabloului poate fi o expresie constantă întregă pozitivă care specifică numărul de elemente. Tipul elementelor este tipul tabloului.

Orice tablou are un nume (care acționează ca un pointer constant la primul element, dar nu este efectiv un pointer). Deci, un nume de tablou este un identificator. Poate fi convenabil să-l considerăm ca un pointer constant în anumite contexte, deoarece comportamentul este în mare parte același și este implicit convertibil către pointer, dar nu este, în sine, un pointer.

2.1. Tablouri unidimensionale

Declarare: `tip nume_tablou[dimensiune];`

- Pentru referirea unui element se folosește operatorul de indexare [], precizând numele tabloului și poziția elementului în tablou (*indexul sau indicele*): `nume_tablou[index]`

2.2. Tablouri multidimensionale

Declarare: `tip nume_tablou[dimens1][dimens2]...[dimensN];`

- Un tablou cu mai multe dimensiuni este de fapt un tablou unidimensional având ca elemente alte tablouri.
- Referirea unui element se face astfel: `nume_tablou[index1][index2]...[indexN];`

2.3. Elemente de tablou

Un element de tablou poate să apară oriunde poate să apară o variabilă simplă cu același tip ca și tipul tabloului : atribuiri, citiri de la consolă, afișări, calcule (expresii), apeluri de funcții.

2.4. Inițializarea tablourilor la declarare

Inițializarea se poate face la declarare, prin utilizarea unei liste de valori separate prin virgulă și delimitată de acolade.

a) tablouri unidimensionale: `tip nume_tablou[dimensiune] = {ec0, ..., eci};`

b) tablouri multidimensionale (matrice): `tip nume_tablou[n][m] = {{ec11, ..., ec1m},
{ec21, ..., ec2m}, ...,
{ecn1, ..., ecnm}};`

2.5. Prelucrarea tuturor elementelor unui tablou

Se utilizează cel mai frecvent instrucțiunea ciclică *for*() standard pentru a prelucra secvențial toate elementele. Variabila de ciclare este folosită ca și index în tablou, iar o altă variabilă sau constantă simbolică ce conține dimensiunea tabloului se folosește în condiția asociată instrucțiunii ciclice. Cnad nu se face distincție între elemente se prefera a folosi o instrucțiune de tip *for_range*, introdusa ulterior în C++.

2.6. Transmiterea tablourilor către funcții

Se poate face prin declararea unui argument de tip tablou, și poate fi fără prima dimensiune. Uzual, se mai transmite și dimensiunile tabloului ca și argument/e separat.

Fiind o formă de apel prin adresă, orice modificare făcută în funcție, a elementelor unui tablou transmis ca parametru, se face asupra tabloului inițial. Dacă însă se folosește modificatorul *const* pentru tablou, atunci acesta este transmis prin valoare și în funcție nu se mai fac modificări asupra elementelor tabloului original.

În cazul tablourilor multidimensionale, prima dimensiune din stanga poate lipsi, celelalte sunt obligatorii. De exemplu:

```
int tab_fcn(char tablou[][25][50]);
```

2.7. Căutarea în tablouri

De obicei se caută poziția unui element cu o anumită valoare.

Dacă tabloul nu este sortat (elementele nu sunt aranjate într-o anumită ordine) se aplică “căutarea liniară” care constă în parcurgerea secvențială (unul după altul) a elementelor tabloului și compararea acestora cu valoarea de căutată. Dacă este găsit un element ce are acea valoare, se returnează indicele elementului, altfel se returnează valoarea -1. Acest algoritm permite găsirea primului element ce are o anumită valoare, însă pot fi mai multe elemente cu aceeași valoare.

2.8. Inserarea în tablouri

În cazul tablourilor nesortate, pentru a insera elemente noi într-un tablou trebuie să ținem cont de următoarele:

- trebuie să existe poziții libere în tablou;
- pentru elementul nou se crează loc prin deplasarea altor elemente, spre dreapta.

În cazul tablourilor sortate trebuie găsită mai întâi poziția în care se va face inserarea.

2.9. Ștergerea din tablouri

Ca și inserarea, ștergerea presupune deplasarea unor elemente, spre stânga.

Se presupune că se cunoaște poziția în care se face ștergerea (dacă nu se face căutarea elementului de tablou cu o anumită valoare).

3. Exemple:

Exemplul 1: program ce citește elementele unui tablou bidimensional, după care determina suma elementelor de pe diagonala principală.

```
#define _CRT_SECURE_NO_WARNINGS
```

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

```
#define MAX 20
```

```
int main(void)
```

```
{
```

```
    int i, j, n, sum = 0, tab[MAX][MAX];
```

```
    printf("\nDimensiunea matricei: ");
```

```
    scanf("%d", &n);
```

```
    if (n <= 0) {
```

```
        printf("\nDimensiunea data este negativa!");
```

```
        return 1;
```

```
    } //end if
```

```
    printf("\nIntroduceti elementele matricei:");
```

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

```
        printf("\n\tLinia %d: \n", i + 1);
```

```

        for (j = 0; j < n; j++) {
            printf("\t\ttab[%d,%d] = ", i + 1, j + 1);
            scanf("%d", &tab[i][j]);
        }
    } //end for
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            if (i == j)
                sum += tab[i][j];
    printf("\nSuma elementelor de pe diagonala principala este: %d\n ", sum);
    return 0;
}

```

Exemplul 2: program pentru determinarea valorii medii a elementelor unui tablou unidimensional.

```

#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

#define DIM 20

float ValMed(int x[], int dim);

int main(void)
{
    int i, dim;
    int x[DIM];
    printf("\nIntroduceti dimensiunea tabloului: ");
    scanf("%d", &dim);
    if (dim > DIM) {
        printf("\n Dimensiune prea mare !");
        return 1;
    }

    printf("\n Introduceti elementele tabloului:\n");
    for (i = 0; i < dim; i++)
    {
        printf("\tx[%d] = ", i);
        scanf("%d", &x[i]);
    }

    printf("\n Valoarea medie este: %.3f\n", ValMed(x, dim));
    return 0;
}

// Functia determina valoarea medie a elementelor unui tablou
float ValMed(int x[], int n)
{
    int i;
    float sum = 0.0f;
    for (i = 0; i < n; i++)
        sum += x[i];
    return sum / n;
}

```

Exemplul 3: program pentru căutarea directă într-un tablou unidimensional.

```

#define _CRT_SECURE_NO_WARNINGS

```

```

#include <stdio.h>

#define DIM 100

int CautDir(int x[], int dim, int val);

int main(void)
{
    int i, dim, val, poz;
    int x[DIM];
    printf("\nIntroduceti dimensiunea tabloului( <= %d): ", DIM);
    scanf("%d", &dim);
    if (dim > DIM) {
        printf("\n Dimensiune prea mare !");
        return 1;
    }

    printf("\n Introduceti elementele tabloului:\n");
    for (i = 0; i < dim; i++)
    {
        printf("tx[%d] = ", i);
        scanf("%d", &x[i]);
    }

    printf("\n Introduceti numarul de cautat: ");
    scanf("%d", &val);
    poz = CautDir(x, dim, val);
    if (poz < 0)
        printf("\n Numarul %d nu a fost gasit in tablou !\n", val);
    else
        printf("\n Numarul %d a fost gasit in pozitia %d.\n", val, poz + 1);
    return 0;
}

// Functie ce cauta o valoare intr-un tablou
// Returneaza pozitia in caz de suuces sau valoarea -1 in caz de esec
int CautDir(int x[], int n, int val)
{
    for (int i = 0; i <= n - 1; i++)
    {
        if (x[i] == val)
            return i;
    }
    return -1;
}

```

Exemplul 4: program pentru inserarea unei valori într-un tablou unidimensional.

```

#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

#define DIM 20

void Inserare(int tab[], int n, int idx, int val);

int main(void)
{
    int i, dim, val, poz;

```

```

int x[DIM];
printf("\nIntroduceti dimensiunea tabloului: ");
scanf("%d", &dim);
if (dim > DIM) {
    printf("\n Dimensiune prea mare !");
    return 1;
}

printf("\n Introduceti elementele tabloului:\n");
for (i = 0; i < dim; i++)
{
    printf("\tx[%d] = ", i);
    scanf("%d", &x[i]);
}

printf("\n Introduceti numarul de inserat: ");
scanf("%d", &val);
printf("\n Introduceti pozitia in care se va face inserarea: ");
scanf("%d", &poz);
if (poz > DIM-2) {
    printf("\n Nu mai este spatiu de inserare in tablou !\n");
    return 0;
}

Inserare(x, dim, poz, val);
dim++;
printf("\nNumerele din tablou :\n");
for (i = 0; i < dim; i++)
    printf("\t%d", x[i]); //end for
printf("\n");
return 0;
} //end main

void Inserare(int arr[], int n, int idx, int val)
{
    if (idx == -1)
        arr[n] = val; // adaugare
    else {
        for (int i = n; i > idx; i--) //deplasare dreapta
            arr[i] = arr[i - 1];
        arr[idx] = val;
    }
}

```

Exemplul 5: program pentru afisarea unui tablou tridimensional din perspectiva xOy.

```

//3D matrix display
#include<stdio.h>

#define Max1 3 //last limit - planes
#define Max2 2 //midle limit - rows
#define Max3 4 //first limit - columns

int main( ) {
    int i, j, k;
    double arr[Max1][Max2][Max3] = {
        {{-0.1, 0.22, 0.3, 4.3}, {2.3, 4.7, -0.9, 2}},
        {{0.9, 3.6, 4.5, 4}, {1.2, 2.4, 0.22, -1}},
    };
}

```

```

    {{8.2, 3.12, 34.2, 0.1}, {2.1, 3.2, 4.3, -2.0}}
};

printf(":::3D Array Elements:::\n\n");
for (i = 0; i < Max1; i++)
{
    for (j = 0; j < Max2; j++)
    {
        for (k = 0; k < Max3; k++)
        {
            printf("%6.2lf\t", arr[i][j][k]);
        }
        printf("\n");
    }
    printf("\n");
}
} //main

```

Exemplul 5: program ce returneaza un tablou static declarat intr-o functie.

```

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define MAX 10

/* function to generate and return random numbers */
int * getRandom( );

/* main function to call the getRandom( ) function */
int main( )
{
    /* a pointer to an int */
    int *p;
    int i;
    p = getRandom( );
    for ( i = 0; i < MAX; i++ )
        printf("\np[ %d] : %d", i, p[i]);
}

int * getRandom( ){
    static int r[MAX];    //must be static
    int i;
    /* set the seed */
    srand( (unsigned)time( NULL ) );
    for(i = 0; i < MAX; ++i)
    {
        r[i] = rand( );    //generate random numbers
        printf( "r[%d] = %d\n", i, r[i]);
    }
    return r;
}

```

4. Întrebări:

- Cum se pot accesa elementele unui tablou ?
- Cum se face inițializarea unui tablou ?

- Ce presupune inserarea, respectiv ștergerea unui element dintr-un tablou ?
- Cum se realizează transmiterea unui tablou într-o funcție?

5. Teme:

1. Scrieți un program pentru determinarea valorii medii a elementelor pozitive/negative dintr-un tablou unidimensional.
2. Scrieți un program pentru determinarea celui mai mic element pozitiv dintr-un tablou unidimensional.
3. Scrieți o aplicație C/C++ care citește de la tastatură un șir de 10 valori întregi. Definiți o funcție care primește șirul ca parametru și apoi îl afișează ordonat crescător.
4. Scrieți o aplicație C/C++ care definește o parolă (în format șir de caractere). Programul citește în mod repetat șirurile de caractere introduse de la tastatură, până când utilizatorul scrie parola corectă. Să se afișeze numărul de încercări până la introducerea parolei corecte.
5. Scrieți o aplicație C/C++ care citește de la tastatură două șiruri de caractere reprezentând numele și prenumele unei persoane. Afișați-le cu majuscula, separate prin două spații de tabulare.
6. Scrieți o aplicație C/C++ care definește două matrici de valori întregi. Dimensiunea și elementele matricilor sunt citite de la tastatură. Scrieți funcțiile care:
 - a) afișează pozițiile elementelor pare din fiecare matrice;
 - b) afișează suma elementelor impare din ambele matrice;
 - c) afișează suma matricelor;
7. Citiți de la tastatură elementele unei matrice cu elemente de tip float, cu dimensiunea 3x3. Rearanjați coloanele matricei astfel ca suma elementelor de pe o anumită coloană să fie mai mică decât suma elementelor de pe coloana următoare.
8. Scrieți un program care primește de la tastatură "n" valori reale într-un tablou unidimensional, calculează cu o funcție valoarea medie a valorilor introduse și afișează cu o altă funcție doar valorile din tablou mai mari decât valoarea medie calculată.
9. Să se scrie o aplicație C/C++ în care se citesc într-un tablou unidimensional "n" valori întregi și se determină numărul elementelor negative impare. Să se afișeze acest număr și elementele respective.
10. Scrieți programul care citește elementele întregi ale unui tablou unidimensional și construiește într-o funcție un alt tablou unidimensional în care se vor stoca resturile împărțirii elementelor primului tablou la numărul elementelor pozitive din acesta.
11. Se citește de la tastatură un șir de caractere. Scrieți funcția care inversează șirul și apoi formează un alt șir de caractere ce va conține elementele de pe pozițiile pare ale șirului inversat. Afișați șirurile obținute.
12. Să se citească de la tastatură elementele întregi ale unei matrice de dimensiune $m \times n$. Dacă matricea este pătratică să se afișeze elementele diagonalei secundare, altfel să se afișeze suma elementelor de pe o coloană dată, c. Valorile m, n și c se citesc de la tastatură și se vor scrie funcții pentru operațiile cerute.
13. Pornind de la exemplul 5 în care se arată modul de parcurgere a unei matrice tridimensionale din perspectiva xOy, să se scrie un program care face parcurgerea matricei considerând perspectiva xOz și yOz. Afișați datele precizând din ce perspectivă au fost parcurse.

5'. Homework

1. Write a C/C++ code, that determines the mean value of the negative/positive elements from a one dimensional array.
2. Write a C/C++ program that determines the value of the smallest positive element of a one dimensional array.
3. Write a C/C++ application that reads from the keyboard an array of 10 integer values. Define and implement a function that receives the array as parameter and then displays its elements, ordered increasingly.
4. Write a C/C++ application that defines a password (as a string of characters). The program reads repeatedly the entries from the keyboard until the user enters the right password. Display the number of trials the user entered wrong passwords.
5. Write a C/C++ application that reads from the keyboard two arrays of characters representing the name and surname of a person. Display them with capital letters, separated by a <Tab> space.

6. Write a C/C++ application that defines two matrices of integer values. The matrices' dimensions and their elements are read from the standard input. Write the functions that display:
 - a) the positions of the odd elements in each matrix;
 - b) the sum of all even elements in both matrices;
 - c) the sum of the two initial matrices;
7. Read from the keyboard the elements of a 3x3 float matrix. Rearrange the columns so that the sum of each column's elements is smaller than the sum of the elements of the next column to the right.
8. Write a program that reads from the keyboard n integer values into a one-dimensional array. Implement a function that calculates their average value. Another function will display the elements from the initial array that are greater than the computed average.
9. Write a program that reads from the keyboard n integer values into a one-dimensional array. Display the number of negative odd elements and their values.
10. Write a program that reads the integer elements of a one-dimensional array. Implement a function that builds another array containing the remainders obtained by dividing the values in the initial array to the number of the positive elements.
11. Read from the keyboard an array of characters. Define the function that reverses the array and populates another array with the elements from the even positions of the reverted array. Display the results.
12. Read from the keyboard the elements of an $m \times n$ integer matrix.
If the matrix is square, display the elements from the secondary diagonal. If not, print the sum of all the elements from a certain column, c .
The values of m , n and c are read from the keyboard. All the operations should be implemented in specific functions.
12. Print every occurrence of each char from the English alphabet in a test-string used in old telegraphy, one/line as a triplet of "the searched char" "the offset inside string where it's found" and "the matched char". Print the string with the first occurrence of every character converted in upper-case.
13. Starting from the example 5 which shows how to display a three-dimensional matrix from the perspective of xOy , write a program that goes through the matrix considering the perspective xOz and yOz . Display the data specifying from which perspective they were browsed.