Ministerul Educaţiei și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Facultatea Calculatoare, Informatică şi Microelectronică

**RAPORT**

Lucrare de laborator Nr.3

*la Programarea Calculatoarelor*

Tema: Prelucrarea tablourilor unidimensionale (vectorilor) în limbajul C

A efectuat: st. gr. SI-212 Miliș Daniel

A verificat: lect. asist. Mititelu Vitalii

Chişinău 2021

**Lucrare de laborator Nr.3**

**Tema:** Prelucrarea tablourilor unidimensionale (vectorilor) în limbajul C

**Scopul:** Studierea posibilităților și mijloacelor limbajului C pentru programarea algoritmilor cu structură ramificată și ciclică la prelucrarea tablourilor unidimensionale.

**Sarcina:** Scrieți un program care citește de la tastatură numărul n>1 de elemente ale tabloului, apoi citește de la tastatură aceste n elemente ale tabloului, efectuează calculele indicate în variantă și afișează pe ecran rezultatul

**Varianta 14:**

Să se determine valoarea minimală dintre elementele tabloului și numărul de elemente cu această valoare, precum și media aritmetică a tuturor elementelor nenule în tablou.

**Rezumat:**

# Ce este un tablou unidimensional?

Tabloul unidimensional este o structura de date careia i se atribuie un nume. Este format dintr-o colectie de elemente de acelasi tip, asezate continuu in memorie. Elementele pot fi accesate individual prin indici. Toate elementele au un predecesor (cu exceptia primului element) si un succesor (cu exceptia ultimului element)

# Cum declar un tablou unidimensional?

Pentru a declara un tablou unidimensional ai nevoie de 3 proprietati: tipul, numele si dimensiunea acestuia. Cateva exemple:

* int V[25]; // un tablou ce retine 25 de numere intregi
* char s[40]; // un tablou ce retine 40 de caractere
* float x[30]; // un tablou ce retine 30 de numere reale

Un element al unui tablou poate fi utilizat ca orice alta variabila. Ca sa accesezi un element dintr-un tablou trebuie sa faci referire la indicele acestuia, trecut intre paranteze drepte. Un aspect important este faptul ca **poti efectua operatii doar asupra unui singur element, nu poti modifica tot tabloul deodata**.

# Initializarea tablourilor unidimensionale

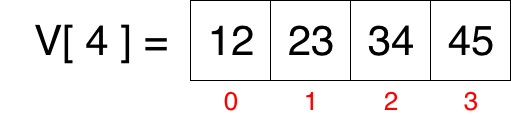
Tablourile, la fel ca si variabilele pot fi initializate in trei moduri:

* Prin introducerea datelor de la tastatura
* Prin declararea acestuia global – toate elementele vor fi zero
* Imediat dupa declararea acestora

(vezi in video cum am intializat pe rand tablourile)

## Accesul elementelor dintr-un tablou unidimensional

* Accesul fiecarui element al tabloului se face prin numele acestuia urmat intre paranteze drepte [ ] de indicele sau (adica pozitia pe care acesta o ocupa in tablou)
* In limbajul C++ , indicii tablourilor incep numaratoarea de la 0 si se termina la N – 1 (unde N este numarul total de elemente)

[](https://tpn-wp.s3.eu-central-1.amazonaws.com/uploads/2019/02/13210221/tablouri-unidimensionale.png)

Primul element din vector are indicele zero.

## Inserarea si stergerea elementelor dintr-un „vector”

### Adaugarea unui element nou

Pentru a adauga un element intr-un tablou unidimensional, tot ceea ce trebuie sa facem este sa marim dimensiunea logica n a vectorului, si sa memoram noul element in casuta ce am eliberat-o.

**int** V[10] = {4, 2, 1, 6};

**int** n = 4;

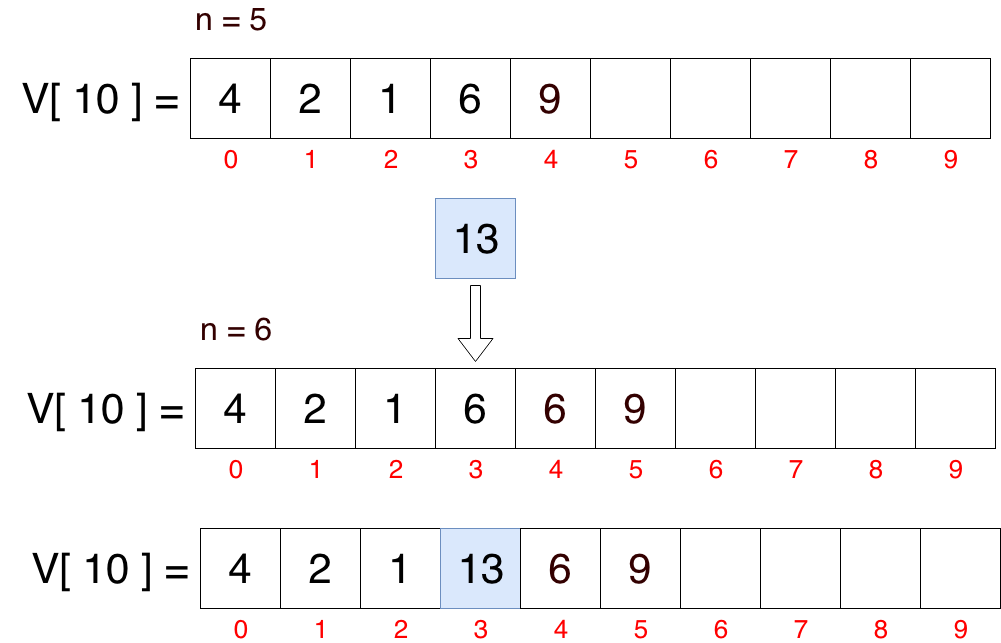
V[n] = numar\_nou;

n++;

//Prescurtat: V[n++] = numar\_nou;

### Inserarea unui element nou

Presupunand ca avem un sir „V” cu „n” elemente, si dorim sa inseram elementul „numar\_nou” pe pozitia „p”.

[](https://tpn-wp.s3.eu-central-1.amazonaws.com/uploads/2019/02/13210223/tablouri-unidimensionale-inserare-element.png)

In imaginea de mai sus, am inserat elementul „13” pe pozitia 3. Cum am procedat?

* Am marit mai intai dimensiunea logica (n) cu un element
* Am parcurs vectorul de la coada „si am tras” toate elementele dupa indicele 3 cu o pozitie la dreapta
* Am inlocuit valoarea din V[3]

**int** V[10] = {4, 2, 1, 6, 9};

**int** n = 5;

n++; //Marim numarul total de elemente

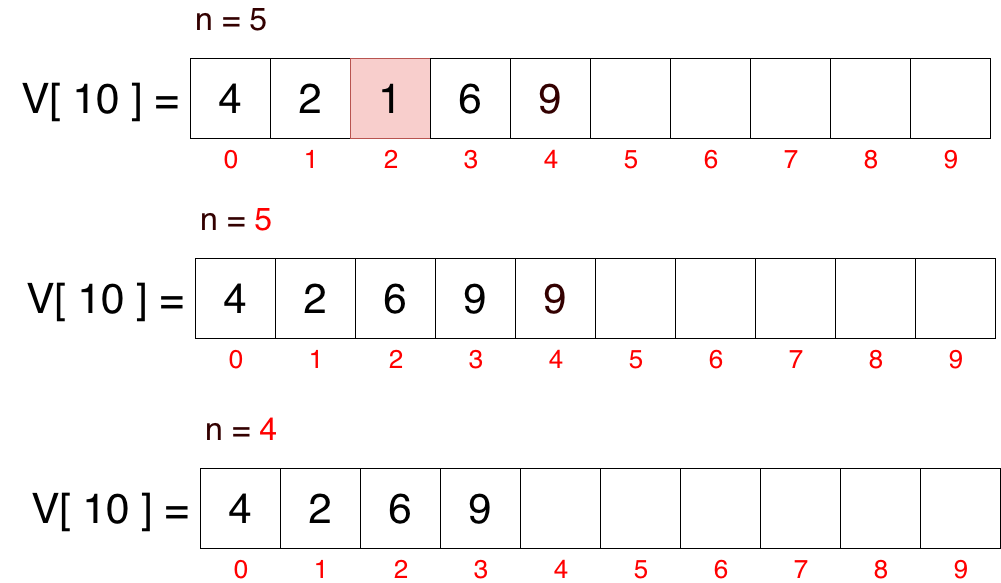
**for**(**int** i = n - 1 ; i > p ; i--)

V[i] = V[i - 1]; //Mutam fiecare element intre p si capat la dreapta

V[p] = numar\_nou; //Inlocuim

### Stergerea unui element

Presupunand ca avem un sir „V” cu „n” elemente, si dorim sa stergem elementul de pe pozitia „p”.

[](https://tpn-wp.s3.eu-central-1.amazonaws.com/uploads/2019/02/13210225/tablouri-unidimensionale-stergere-element.png)

In imaginea de mai sus am sters elementul de pe pozitia 2. Cum am procedat?

* Am parcurs vectorul incepand de pe pozitia „p”
* Am tras toate elementele la stanga
* Am micsorat n cu o unitate

**int** V[10] = {4, 2, 1, 6, 9};

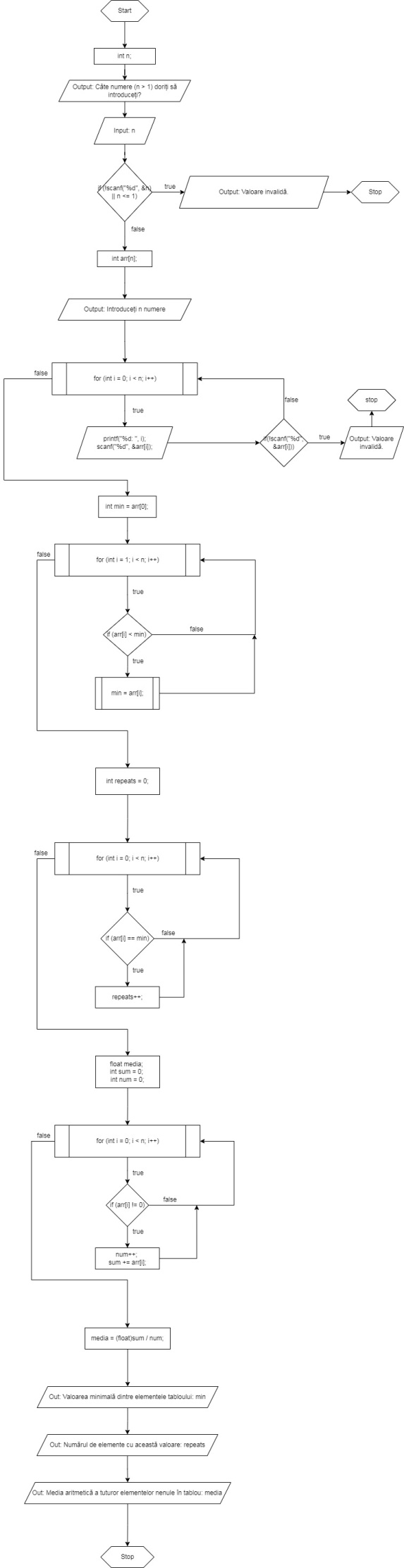
**int** n = 5;

**for**(**int** i = p ; i < n - 1; i ++) // Parcurgem de la p

V[i] = V[i+1]; // Tragem toate elementele la stanga

n--; //Micsoram n cu o unitate

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

****

**Codul deplin al programului:**

#include <stdio.h>

int main () {

    int n;

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

    if (!scanf("%d", &n) || n <= 1) {

        printf ("Valoare invalidă.\n");

        return (1);

    }

    int arr[n];

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

    for (int i = 0; i < n; i++){

        printf("%d: ", i);

        if(!scanf("%d", &arr[i])) {

            printf ("Valoare invalidă.\n");

            return (1);

        }

    }

    printf("\n");

    int min = arr[0];

    for (int i = 1; i < n; i++) {

        if (arr[i] < min) {

            min = arr[i];

        }

    }

    int repeats = 0;

    for (int i = 0; i < n; i++) {

        if (arr[i] == min) {

            repeats++;

        }

    }

    float media;

    int sum = 0;

    int num = 0;

    for (int i = 0; i < n; i++) {

        if (arr[i] != 0) {

            num++;

            sum += arr[i];

        }

    }

    media = (float)sum / num;

    printf("Valoarea minimală dintre elementele tabloului: %d\n", min);

    printf("Numărul de elemente cu această valoare: %d\n", repeats);

    printf("Media aritmetică a tuturor elementelor nenule în tablou: %f\n", media);

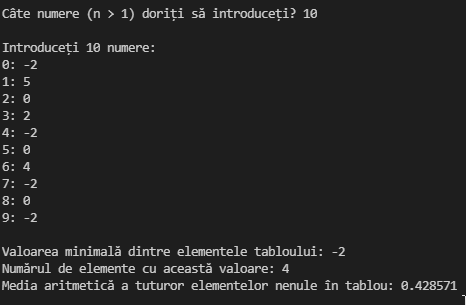
    printf("\n");

    return (0);

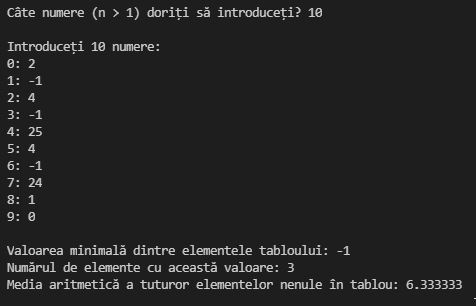
}

**Execuția programului:**

* Set de date nr. 1:



* Set de date nr. 2:



* Set de date nr. 3:

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

**Concluzii:**

1. Pe când alte tipuri de date simple, precum char și int, sunt compuse dintr-o singură valoare, tablourile sunt compuse din mai multe, ordonate și indexate, ceea ce permite stocarea lor într-un singur element, fapt care duce la gestionarea mult mai eficientă a unei cantități mari de date.
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. Există o multitudine de funcții destinate special pentru tablouri, lucru care facilitează efectuarea unor operații specifice cu parametri similari, de mai multe ori, mult mai ușor și mai citeț.

**Surse:**

* **tutoriale-pe.net, curs la C++.** Capitolul: Tablouri unidimensionale (vectori) in C++

https://tutoriale-pe.net/tablouri-unidimensionale-vectori-in-c/