Ministerul Educaţiei, Culturii și Cercetării
Universitatea Tehnică a Moldovei

Departamentul Ingineria Software și Automatică

RAPORT

Lucrarea de laborator nr. 3
la Programarea Calculatoarelor
Varianta 18

A efectuat:
st. gr. TI-206 Cătălin Pleșu
A verificat:
Lector universitar Vitalie Mititelu

Chișinău - 2020

Lucrarea 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 18. Să se determine valorile ultimului element minimal negativ și a primului element maximal negativ, și pozițiile acestora în tablou.

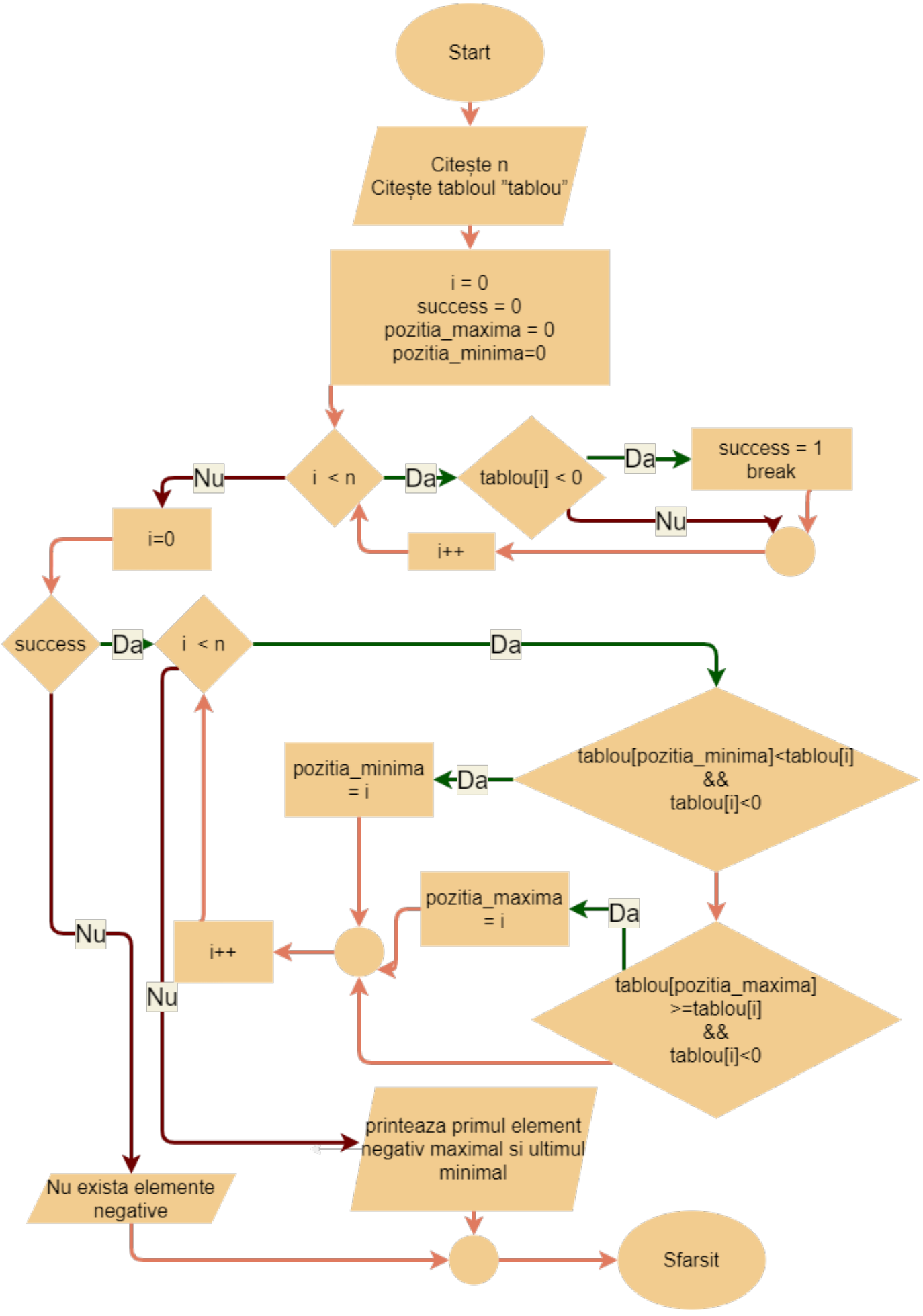
Rezumat succint la temă :

- Pentru a îndeplini sarcina dată a trebuit să utilizez un tablou unidimensional. Când lucrăm cu tablouri în **C** este important să ne asigurăm că operăm doar în limitele tabloului fiind că în caz contrar vom opera cu memorie care are alt scop, poate conține date de orice tip și vom obține date eronate însă nu erori fiind că **C** nu verifică dacă noi ieșim din limitele tabloului.
- În C mărimea tabloului trebuie indicată înainte de a utiliza variabila respectivă sau poate fi modificată cu ajutorul funcției **realloc(ptrointer, biți)**.
- În primul rând am citit o variabilă **n** care care reprezintă lungimea caracterului.
- Apoi cu ajutorul unui ciclu **for** am citit fiecare element al tabloului **tablou.**
- Programul dat trebuie să interacționeze doar cu numerele negative ale tabloului de aceea cu alt ciclu **for** verific dacă cel puțin un element este negativ, în caz contrar se va afișa la ecran „**condiții nesatisfacute**”.
- Apoi cu alt ciclu **for** se va verifica dacă elementul **i** este primul element maxim sau ultimul element minim.
- În acest program am utilizat funcțiile : main(), scanf() și printf(); ciclul for(; ;); și instrucțiunea condițională if();
- Acest program are o structură destul de simplă , având doar funcția **main** , programul nu a avut nevoie de funcții suplimentare, structuri, uniuni sau pointeri cu excepția tabloului.
- Acest program este dependent de biblioteca **stdio.h** care permite citirea și afișarea pe ecran.

Resurse utilizate pentru realizarea lucrării de laborator:

- Visual Studio – editor de text.
- MinGW – conține compilatoru gcc.
- LibreOffice – crearea documentului lucrării.
- Draw.io – pentru crearea schemei logice.

Schema logică a algoritmului :

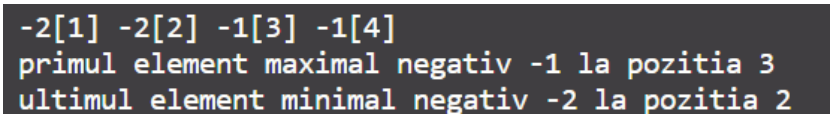
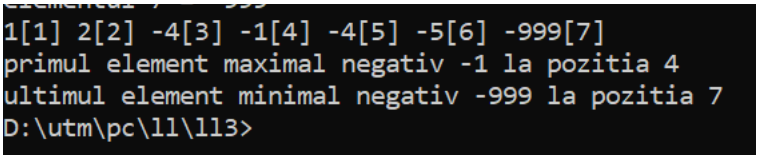
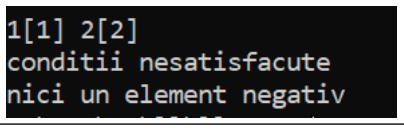


Cod sursă in limbajul C :

```
#include <stdio.h> //varianta 18
int main(void)
{
    printf("\n\n\t\t\t");
    printf("Lucrarea de laborator nr. 3 Varianta 18\n\n\n");
    printf("de la tastatura se va citi numarul n \ncare reprezinta numarul de elemente al ta
bloului\nn= ");
    int n;
    scanf("%d", &n);
    int tablou[n];
    int success = 0;
    for (int i = 0; i < n; i++)
    {
        printf("\nelementul %d = ", i + 1);
        scanf("%d", &tablou[i]);
    }
    //printeaza elementele tabloului cu pozitia lor in format mai usor de inteles pentru oam
eni n+1
    for (int i = 0; i < n; i++)
    {
        printf("%d[%d] ", tablou[i], i + 1);
    }
    int firstspossitionmax = 0;
    int lastpossitionmin = 0;
    for (int i = 0; i < n; i++)
    { // cautam cel putin un element negativ pentru functionarea corecta a programului
        if (tablou[i] < 0)
        {
            success = 1;
            break; //pentru eficienta
        }
    }
    if (success)
    {
        for (int i = 0; i < n; i++)
        {
            if (tablou[firstspossitionmax] > 0)
            {
                firstspossitionmax = i;
            }
        }
        for (int i = 0; i < n; i++)
        {
            //selecteaza pozitia primului element maxim
            if ((tablou[firstspossitionmax] < tablou[i]) && (tablou[i] < 0))
            {
                firstspossitionmax = i;
            }
            //selecteaza pozitia ultimului element minimal
            if ((tablou[lastpossitionmin] >= tablou[i] && (tablou[i] < 0)))
            {
```

```
            lastpossitionmin = i;
        }
    }
    printf("\nprimul element maximal negativ %d la pozitia %d\nultimul element minimal negativ %d la pozitia %d", tablou[firstspossitionmax], firstspossitionmax + 1, tablou[lastpossitionmin], lastpossitionmin + 1);
  }
  else
  {
    printf("\nconditii nesatisfacute\nnici un element negativ");
  }
  return 0;
}
```

Verificarea datelor de ieșire :

Nr.	Date de intrare	Rezultat
1	n= 4 elementul 1 = -2 elementul 2 = -2 elementul 3 = -1 elementul 4 = -1	
2	n= 7 elementul 1 = 1 elementul 2 = 2 elementul 3 = -4 elementul 4 = -1 elementul 5 = -4 elementul 6 = -5 elementul 7 = -999	
3	n= 2 elementul 1 = 1 elementul 2 = 2	

Analiza datelor de ieșire :

- În caz că introducem doar numere pozitive, programul va afișa pe ecran că, condițiile sunt nesatisfăcute, adică că nici un element nu este negativ.
- În caz că avem cel puțin un element negativ programul va face operațiile necesare și ne va afișa răspunsul dorit.

Concluzii :

Am studierea posibilităților și mijloacelor limbajului C pentru programarea algoritmilor cu structură ramificată și ciclică la prelucrarea tablourilor unidimensionale. Am reușit să scriu un program ce operează cu un tablou unidimensional. Am utilizat condițiile și ciclurile pe care le știam de mai devreme. În C spre deosebire de JavaScript este mai dificil de operat cu matrici și tablouri însă C are alte implemetnări, care necesită o performanță ridicată spre deosebire de JS. O parte esențială a rezolvării oricărei probleme este să înțelegem condiția acestei probleme. Ar fi o idee bună să planific codul, inclusiv cu schema bloc astfel nu voi ajunge să reâncep un program de la zero. Este mult mai clar să urmărești o schemă logică decât codul sursă dacă acesta nu este scris conform celor mai bune practici.