Ministerul Educaţiei, Culturii și Cercetării al Republicii Moldova

Universitatea Tehnică a Moldovei

Catedra Informatica si Ingineria Sistemelor

**RAPORT**

Lucrarea de laborator nr.4

la Programarea Calculatoarelor

A efectuat:

st. gr. C-171 D. Melniciuc

A verificat:

dr., conf.univ. M. Kulev

Chişinău 2017

**Lucrarea de laborator nr.4**

**Tema:** **Tablouri liniare**

**Scopul lucrarii practice:**

Studierea şi utilizarea instrucţiunilor de bază ale limbajului pentru prelucrarea şi modificarea masivelor unidimensionale

**Notiuni de baza:**

Numim tablou o colecţie (grup, mulţime ordonată) de date, de acelaşi tip, situate într-o zonă de memorie continuă. Tablourile sunt variabile compuse (structurate), deoarece grupează mai multe elemente. Variabilele tablou au nume, iar tipul tabloului este dat de tipul elementelor sale. Elementele tabloului pot fi referite prin numele tabloului şi indicii (numere intregi) care reprezintă poziţia elementului în cadrul tabloului.

În funcţie de numărul indicilor utilizaţi pentru a referi elementele tabloului, putem întâlni tablouri unidimensionale (vectorii) sau multidimensionale (matricele sunt tablouri bidimensionale).

Ca şi variabilele simple, variabilele tablou trebuie declarate înainte de utilizare.

Modul de declarare:

**tip nume\_tablou[dim\_1][dim\_2]...[dim\_n];**

unde:

tip reprezintă tipul elementelor tabloului (int, float sau char);

nume\_tablou reprezintă numele tabloului;

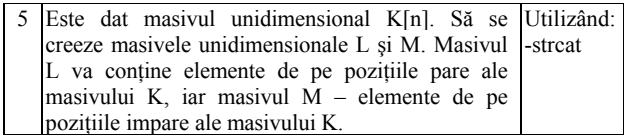
dim\_1,dim\_2,...,dim\_n sunt numere întregi sau expresii constante intregi care reprezinta limitele superioare ale indicilor tabloului.

**Condiţia problemei (sarcina de lucru):**

Să se scrie un program ce va prelucra un masiv

bidimensional cu n linii şi m coloane, având următoarele condiţii:

**Varianta 5:**



**Mersul lucrarii:**

[2]Pentru a programa la calculator e necesara urmarirea unui algoritm strict. Pasii sunt urmatorii:

-cunoasterea problemei;

Citirea si intelegerii corecte a problemei.

-analiza datelor;

Studierea datelor problemei pentru a destinge datele de tip input, output si date intermediare.

-elaborarea algoritmului prin schema logica;

Elaborarea schemei logice a programului.

-codul (textului) programei;

Scrierea codului propriu-ziz.

-testarea;

Rularea problemei pentru a determina erori logice sau de sintaxa.

-verificarea rezultatelor

Introducerea diferitor date pentru a verifica mai multe posibilitati.

-analiza rezultatelor si formarea concluziilor;

Formare concluziilor adecvate dupa efectuarea pasilor anteriori.

Analiza datelor:

In problema data am urmatoarele conditii

Să se

creeze masivele unidimensionale L şi M. Masivul

L va conţine elemente de pe poziţiile pare ale

masivului K, iar masivul M – elemente de pe

poziţiile impare ale masivului K.

1. Creem masivul principal(K);
2. Introducerea elementelor de pe pozitiile pare din masivul principal(K) in alt masiv(L);
3. Introducerea elementelor de pe pozitiile impare din masivul principal(K) celalalt masiv(M)

**Schema logica:**

“Introducem lungimea vectorului”

n

“Incarcam valori”

i=0; i<n; i++

i=0; i<n; i++

i=0; i<n; i++

i%2==0

i%2!=0

“introducem k[i]”

M[i2] = K[i]

L[i1] = K[i]

i2++

i1++

k[i]

“Afisam vector”

M[i]

i=0; i<n; i++

Elementele de pe pozitii pare a lui k:

i2 = 0;

L[i]

i=0; i<n; i++

Elementele de pe pozitii impare a lui k:

i1 = 0;

k[i]

i=0; i<n; i++

“k = “

**Codul (textul) programului in limbajul C:**

#include <stdio.h>

#include <conio.h>

#include <math.h>

#include <stdlib.h>

int main()

{

int i, n, k[20],m[20], l[20];

int i1, i2;

printf("lungimea vectorului\n");

scanf("%d", &n);

printf("Incarcam valori\n");

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

{

printf("introducem k[%d]=", i+1);

scanf("%d",&k[i]);

}

printf("\nAfisam vectoru\n");

printf("k=");

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

printf("%3d\t", k[i]);

i1=0;

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

{

if(i%2==0)

{

l[i1]=k[i];

i1++;

}

}

printf("\nElementele de pe pozitii impare a lui k: ");

for(i=0;i<i1;i++)

{

printf("%3d\t", l[i]);

}

i2=0;

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

{

if(i%2!=0)

{

m[i2]=k[i];

i2++;

}

}

printf("\nElementele de pe pozitii pare a lui k: ");

for(i=0;i<i2;i++)

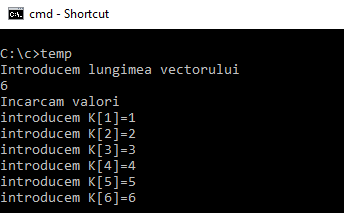
{

printf("%3d", m[i]);

}

return 0;

}

**Rezultatele obtinute:**

Introducerea lungimii vectorului

Apoi introducerea valorilor



Afisarea vectorului

Indeplinirea sarcinii

**Bibliografie:**

[1] Derevlenco, Vasile și alț. Programare : Indicaţii de laborator. Partea 1. – Ch.: Tehnica – UTM, 2014. <http://library.utm.md>

[2]Conspectul si notele de la lectiile teoretice

**Concluzie:**

In urma efectuarii programului si a rularii acestuia au fost obtinute anumite abilitati in domeniu programarii si anume studierea tabelelor unidimensionale (1D).

**Avantaje**

Putem da orice numarul de linii ale vectorului.

Deasemenea putem da valoare pentru fiecare pozitie a vectorului.

Exista comunicare indirecta a progamatorului si a clientului datorita faptului ca programul e interactiv.

**Dezavantje**

Programul dat doar depisteaza elemente de pe poziţiile pare si impare.

**Dezvoltarea programului**

Sa include pozitiile impare si pare doar int-un masiv, nu doua.

Micsorarea numarului liniilor din cod.

Restructurarea codului pentru un nivel de citire mai usor.