**Universitatea Tehnică a Moldovei**

**Facultatea *Calculatoare, Informatică și Microelectronică***

**Specialitatea *Tehnologii Informaționale***

A blue and black logo

Description automatically generated

Raport

**la lucrarea de laborator nr. 2**

**Tema: *“Algoritmi de prelucrare a structurilor, a tablourilor și pointeri la structuri”***

**Disciplina: “Structuri de date și algoritmi”**

Varianta 4

|  |  |  |
| --- | --- | --- |
| **A efectuat:** | Student grupa TI-231 FR | Apareci Aurica |
| **A verificat:** | Asistent universitar | Mantaluță Marius |

**Chișinău 2024**

**Cuprins**

[**1.** **Cadrul teoretic** 3](#_Toc166454489)

[**2.** **Repere teoretice** 3](#_Toc166454490)

[**3.** **Listingul programului** 3](#_Toc166454491)

[**4.** **Testarea aplicației** 8](#_Toc166454492)

[**5.** **Concluzii** 9](#_Toc166454493)

# **Cadrul teoretic**

**Scopul:** Programarea algoritmilor de prelucrare a structurilor și a tablourilor de la structuri prin utilizarea funcțiilor, pointerilor, alocării dinamice a memoriei în limbajul C.

**Sarcina:** Pentru tabloul unidimensional dat cu elemente de tip structură (conform variantelor) să se afișeze la ecran următorul meniu recursiv de opțiuni:

1. Introducerea elementelor tabloului de la tastatură.

2. Afișarea elementelor tabloului la ecran.

3. Sortarea tabloului crescător/descrescător(2 tehnici de sortare la alegere).

0. Ieșire din program.

Să se elaboreze funcțiile pentru implementarea opțiunilor meniului.

**Varianta 4.** Structura *Cinematograf* cu câmpurile: denumirea, adresa, telefonul, numărul de locuri, prețul biletelor.

# **Repere teoretice**

**Structurile** sunt un tip de date compus care permite gruparea diferitor tipuri de date sub o singură entitate logică într-un singur bloc de memorie. Ele sunt folosite pentru a reprezenta obiecte complexe, cum ar fi persoane, produse, sau date specifice unei aplicații.

Structurile pot fi de două tipuri: **Structuri fără nume:** Aceste structuri nu sunt asociate cu un nume specific. Ele sunt folosite atunci când este necesară declararea unei variabile globale/instanțe de tip structură. **Structuri cu nume:** Aceste structuri sunt asociate cu un nume specific. Ele pot fi folosite în program pentru a declara variabile și pentru a accesa membrii structurii.

# **Listingul programului**

#include <stdio.h>

#include <stdlib.h>

**main.c**

#include "user.h"

**int main()**{

while (go)

{

userChose = Menu();

BL();

}

return 0;

}

#include <stdio.h>

**struct.h**

#include <stdlib.h>

#include <ctype.h>

**typedef struct Cinematograf {**

char denumirea[50];

char adresa[100];

char telefonul[15];

int numarLocuri;

float pretBilete;

} Cinematograf;

**Cinematograf \* ReadData()**{

Cinematograf \*c = (Cinematograf \*)malloc(sizeof(Cinematograf));

printf("Denumirea: "); fflush(stdin);

scanf("%s", c->denumirea);

printf("Adresa: "); fflush(stdin);

scanf("%s", c->adresa);

printf("Telefonul: "); fflush(stdin);

scanf("%s", c->telefonul);

printf("Numarul de locuri: "); fflush(stdin);

scanf("%d", &c->numarLocuri);

printf("Pretul biletelor: "); fflush(stdin);

scanf("%f", &c->pretBilete);

return c;

}

**void PrintData(Cinematograf \*c)**{

printf("\n--------------------------------\n");

printf("Denumirea: %s\n", c->denumirea);

printf("Adresa: %s\n", c->adresa);

printf("Telefonul: %s\n", c->telefonul);

printf("Numarul de locuri: %d\n", c->numarLocuri);

printf("Pretul biletelor: %.2f\n", c->pretBilete);

printf("\n--------------------------------\n");

}

**user.h**

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include "struct.h"

Cinematograf \* \_cinematografe; int nr = 0;

**void Read()**{

printf("Dati numarul de cinematografe ce urmeaza a fi introduse: ");

scanf("%d", &nr);

\_cinematografe = (Cinematograf \*)malloc(nr \* sizeof(Cinematograf));

for (int i = 0; i < nr; i++)

{

printf("Cinematograf %d\n--------------------------\n", i + 1);

Cinematograf \* c = ReadData();

\_cinematografe[i] = \*c;

}

}

**void Show()**{

for (int i = 0; i < nr; i++)

{

PrintData(&\_cinematografe[i]);

}

}

**void ChooseSortMethod(){**

int optiune = 0;

printf("1. Sortare crescatoare bubbleSort\n");

printf("2. Sortare descrescatoare bubbleSort\n");

printf("3. Sortare crescatoare insertionSort\n");

printf("4. Sortare descrescatoare insertionSort\n");

printf("Alegeti o optiune: ");

scanf("%d", &optiune);

if (optiune == 1) {

BubbleSortCrescator();

} else if (optiune == 2) {

BubbleSortDescrescator();

} else if (optiune == 3) {

insertionSortCrescator(\_cinematografe, nr);

} else if (optiune == 4) {

insertionSortDescrescator(\_cinematografe, nr);

} else {

printf("Optiune invalida!\n");

}

}

**void BubbleSortCrescator()**{

int i, j;

Cinematograf temp;

for (i = 0; i < nr - 1; i++)

{

for (j = 0; j < nr - i - 1; j++)

{

if (\_cinematografe[j].pretBilete > \_cinematografe[j + 1].pretBilete)

{

temp = \_cinematografe[j];

\_cinematografe[j] = \_cinematografe[j + 1];

\_cinematografe[j + 1] = temp;

}

}

}

}

**void BubbleSortDescrescator()**{

int i, j;

Cinematograf temp;

for (i = 0; i < nr - 1; i++)

{

for (j = 0; j < nr - i - 1; j++)

{

if (\_cinematografe[j].pretBilete < \_cinematografe[j + 1].pretBilete)

{

temp = \_cinematografe[j];

\_cinematografe[j] = \_cinematografe[j + 1];

\_cinematografe[j + 1] = temp;

}

}

}

}

**void insertionSortCrescator(Cinematograf array[], int size) {**

for (int step = 1; step < size; step++) {

Cinematograf key = array[step];

int j = step - 1;

while (j >= 0 && key.pretBilete < array[j].pretBilete) {

array[j + 1] = array[j];

--j;

}

array[j + 1] = key;

}

}

**void insertionSortDescrescator(Cinematograf array[], int size) {**

for (int step = 1; step < size; step++) {

Cinematograf key = array[step];

int j = step - 1;

while (j >= 0 && key.pretBilete > array[j].pretBilete) {

array[j + 1] = array[j];

--j;

}

array[j + 1] = key;

}

}

int userChose = 0;

int go = 1;

**int Menu()**{

printf("-------------------------Meniul Aplicatiei-------------------------\n");

printf("1 . \tCitire cinematografe\n");

printf("2 . \tAfisare cinematografe\n");

printf("3 . \tSortare cinematografe\n");

printf("4 . \tElibereaza memoria\n");

printf("0 . \tExit\n");

printf("---------------------------------------------------------------------\nOptiunea aleasa --> ");

int op;

scanf("%d", &op);

system("cls");

return op;

}

**void PressAnyKey()**{

printf("\nAtingeti o tasta pentru a continua\n");

getch();

system("cls");

}

**void BL()**{

switch (userChose)

{

case 1:

{

Read();

PressAnyKey();

} break;

case 2:

{

Show();

PressAnyKey();

}break;

case 3:

{

ChooseSortMethod();

PressAnyKey();

}break;

case 4:

{

free(\_cinematografe);

nr = 0;

PressAnyKey();

}break;

case 0:

{

go=0;

system("cls");

printf("Aplicatia s-a oprit cu succes");

getch();

}break;

default:printf("Optiune necunoscuta\nIncercati din nou");break;

}

}

# **Testarea aplicației**

A black screen with white text

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Nr. | Input | Output |
| 1. | Citire cinematografe |  |
| 2. | Afișare cinematografe |  |
| 3. | Sortare cinematografe |  |

A screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer

Description automatically generated

Fig. 4 Afișarea structurilor sortate descrescător utilizând **InsertionSort**

Fig. 3 Afișarea structurilor sortate crescător utilizând **InsertionSort**

Fig. 2 Afișarea structurilor sortate descrescător utilizând **BubbleSort**

Fig. 1 Afișarea structurilor sortate crescător utilizând **BubbleSort**

# **Concluzii**

În concluzie, această lucrare de laborator a fost abordată cu succes, atingând obiectivele propuse. Am dezvoltat o aplicație în limbajul C pentru manipularea structurilor de tip tablou unidimensional, focalizându-ne pe realizarea unei funcționalități complexe, care include introducerea, afișarea și sortarea cinematografelor.

Această lucrare de laborator s-a concentrat pe utilizarea și gestionarea eficientă a structurilor în limbajul C. Structurile sunt entități de date complexe care permit gruparea mai multor tipuri de date sub o singură denumire. În cazul nostru, am definit o structură numită "Cinematograf" pentru a stoca informații despre fiecare cinematograf, cum ar fi denumirea, adresa, telefonul, numărul de locuri și prețul biletelor. Prin definirea și utilizarea structurilor, am putut organiza și gestiona informațiile într-un mod coerent, transformând o serie de variabile izolate într-o entitate mai complexă și ușor de gestionat.

Prin integrarea structurilor în codul nostru, am demonstrat avantajele organizării și gestionării eficiente a datelor în programele C, contribuind la dezvoltarea abilităților noastre de programare și la înțelegerea practică a conceptelor fundamentale ale limbajului C.