

Ministerul Educației, Culturii și Cercetării
Universitatea Tehnică a Moldovei



Departamentul Ingineria Software și Automatică

RAPORT

Lucrarea de laborator nr. 1
la Structuri de date și algoritmi
Varianta 18

A efectuat:
st. gr. TI-206
A verificat:
Lector universitar

Cătălin Pleșu
Vitalie Mititelu

Lucrarea de laborator nr. 1

Tema : Structuri și tablouri de la structuri

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 de opțiuni:

1. Alocarea dinamică a memoriei pentru tabloul de structuri.
2. Introducerea elementelor tabloului de la tastatură.
3. Afișarea elementelor tabloului la ecran.
4. Adăugarea unui element nou la sfârșit.
5. Modificarea elementului tabloului.
6. Căutarea elementului tabloului.
7. Sortarea tabloului.
8. Eliminarea elementului indicat din tablou.
9. Eliberarea memoriei alocate pentru tablou.
0. Ieșire din program.

Să se elaboreze funcțiile pentru implementarea opțiunilor meniului.

Varianta 18. Structura Imobil cu câmpurile: proprietarul, tipul, adresa, suprafața, costul.

Rezumat succint la temă :

- În acest program am utilizat o structură definită de mine, aceasta structura are cinci elemente și are dimensiunea de 20 de byte fără alocarea memoriei pentru șirurile de caracter, datorită faptului că am utilizat pointeri la char, programul nu va alocă memorie care să nu fie utilizată.
- Prima funcție **Menu(int *m)** afișează meniul programului și citește opțiunea numerică în adresa variabilei din funcția **main**.
- Funcția **Creation** este un pointer de tipul structurii în interiorul căreia se citește numărul de elemente ale structurii. Se alocă memoria necesară și se returnează acest pointer.
- Funcția **Demo** este o funcție suplimentară care umple tabloul cu date mai mult sau mai puțin aleatorii.
- Restul funcțiilor au niște nume destul de intuitive.
- Funcțiile sunt apelate într-un switch în funcția principală.
- Tabloul și mărimea lui sunt variabile globale.

Cod sursă în limbajul C :

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <math.h>

typedef struct Imobil
{
    char *proprietar;
    char *tip;
    char *adresa;
```

```

float suprafata;
float costul;
} IB;
IB *imobile;
int n;
const char name[][30] = {"Catalin", "Dlinii", "Rita", "Alex", "Ion B", "R Leta", "R Sam", "Cristian",
"danielL", "ioneltuc", "Maria", "Marius_01k"};
const char type[][30] = {"casa", "hotel", "spalatorie auto", "castel", "palat", "gradina zoo", "restau
rant", "apartament", "pestera"};
const char addres[][30] = {"livezilor 12", "eminescu 23", "bul. moscovei 11", "studentilor 7/1", "st
rada 1", "strada 2", "starda 3", "botanica 23"};
void Menu(int *m)
{
    printf("\n1. Alocarea dinamica a memoriei pentru tabloul de structuri.");
    printf("\n2. Introducerea elementelor tabloului de la tastatura. Sau 999 pentru Demo");
    printf("\n3. Afisarea elementelor tabloului la ecran.");
    printf("\n4. Adaugarea unui element nou la sfarsit.");
    printf("\n5. Modificarea elementului tabloului.");
    printf("\n6. Cautarea elementului tabloului.");
    printf("\n7. Sortarea tabloului.");
    printf("\n8. Eliminarea elementului indicat din tablou.");
    printf("\n9. Eliberarea memoriei alocate pentru tablou");
    printf("\n0. Iesire din program.\n");
    fflush(stdin);
    scanf("%d", m);
}

IB *Creation()
{
    printf("Dati numarul de imobile : ");
    scanf("%d", &n);
    if (n < 0)
    {
        while (n < 0)
        {
            printf("nu putem avea un numar negativ de imobile\ndati n : ");
            scanf("%d", &n);
        }
    }

    IB *imobile;
    imobile = (IB *)malloc(n * sizeof(IB));
    return imobile;
}

void Demo()
{
    srand(time(NULL));
    char str[255];
    for (int i = 0; i < n; i++)
    {
        strcpy(str, name[rand() % 12]);
    }
}

```

```

        imobile[i].proprietar = (char *)malloc((strlen(str) + 1) * sizeof(char));
        strcpy(imobile[i].proprietar, str);
        strcpy(str, type[rand() % 9]);
        imobile[i].tip = (char *)malloc((strlen(str) + 1) * sizeof(char));
        strcpy(imobile[i].tip, str);
        strcpy(str, adres[rand() % 8]);
        imobile[i].adresa = (char *)malloc((strlen(str) + 1) * sizeof(char));
        strcpy(imobile[i].adresa, str);
        imobile[i].suprafata = ((float)(rand() % 300) + (float)11 / ((rand() % 9) + 1)) + 16;
        imobile[i].costul = imobile[i].suprafata * (rand() % 21) / ((rand() % 16) + 1) * 1000 + imobile
[i].suprafata * (rand() % 500);
    }
    int totalMem = sizeof(IB) * n;
    for (int i = 0; i < n; i++)
    {
        totalMem += strlen(imobile[i].proprietar);
        totalMem += strlen(imobile[i].tip);
        totalMem += strlen(imobile[i].adresa);
    }

    printf("\nmemoria utilizata de matrice %d B\n", totalMem);
}
void ScanStruct(int i)
{
    char str[255];
    printf("Imobilul %d\n", i + 1);
    printf("proprietar: ");
    fflush(stdin);
    gets(str);
    imobile[i].proprietar = (char *)malloc((strlen(str) + 1) * sizeof(char));
    strcpy(imobile[i].proprietar, str);
    printf("tip: ");
    fflush(stdin);
    gets(str);
    imobile[i].tip = (char *)malloc((strlen(str) + 1) * sizeof(char));
    strcpy(imobile[i].tip, str);
    printf("adresa: ");
    fflush(stdin);
    gets(str);
    imobile[i].adresa = (char *)malloc((strlen(str) + 1) * sizeof(char));
    strcpy(imobile[i].adresa, str);
    printf("suprafata : ");
    scanf("%f", &imobile[i].suprafata);
    printf("costul : ");
    scanf("%f", &imobile[i].costul);
}
void scanMatrix()
{
    printf("\nCitirea imobilelor\n");
    for (int i = 0; i < n; i++)
    {
        ScanStruct(i);
    }
}

```

```

    }
}
void printMatrix()
{
    if (n)
    {
        printf("| Nr.\\t\\tProprietar\\t| \\t\\tip\\t | \\tadresa\\t | \\tsuprafata | costul |\\n");
        for (int i = 0; i < n; i++)
        {
            printf("|%3d |%25s |%25s |%25s |%12.2f m^2 |%12.2f $ |\\n", i + 1, imobile[i].proprietar, i
mobile[i].tip, imobile[i].adresa, imobile[i].suprafata, imobile[i].costul);
        }
    }
    else
    {
        printf("Matricea este goala\\n");
    }
}
void Order()
{
    int option;
    printf("1. sortare dupa pret descrescator\\n2. sortare dupa pret crescator\\n3.sortare dupa sup
rafata descrescatoare\\noricce alt numar. sortare dupa suprafata crescatoare");
    scanf("%d", &option);
    IB temp;
    int sortat;
    switch (option)
    {
    case 1:
        do
        {
            sortat = 1;
            for (int i = 0; i < n - 1; i++)
                if (imobile[i].costul < imobile[i + 1].costul)
                {
                    sortat = 0;
                    temp = imobile[i];
                    imobile[i] = imobile[i + 1];
                    imobile[i + 1] = temp;
                }
        } while (sortat == 0);
        break;
    case 2:
        do
        {
            sortat = 1;
            for (int i = 0; i < n - 1; i++)
                if (imobile[i].costul > imobile[i + 1].costul)
                {
                    sortat = 0;
                    temp = imobile[i];
                    imobile[i] = imobile[i + 1];

```

```

        imobile[i + 1] = temp;
    }
} while (sortat == 0);
break;
case 3:
do
{
    sortat = 1;
    for (int i = 0; i < n - 1; i++)
        if (imobile[i].suprafata < imobile[i + 1].suprafata)
        {
            sortat = 0;
            temp = imobile[i];
            imobile[i] = imobile[i + 1];
            imobile[i + 1] = temp;
        }
} while (sortat == 0);
break;

default:
do
{
    sortat = 1;
    for (int i = 0; i < n - 1; i++)
        if (imobile[i].suprafata > imobile[i + 1].suprafata)
        {
            sortat = 0;
            temp = imobile[i];
            imobile[i] = imobile[i + 1];
            imobile[i + 1] = temp;
        }
} while (sortat == 0);
break;
}
printf("Tabloul a fost sortat cu succes!\n");
}
void Insert()
{
    IB *templmobile = realloc(imobile, ++n * sizeof(IB));
    if (templmobile)
    {
        imobile = templmobile;
    }
    else
    {
        printf("nu e posibil de inserat un element nou");
    }
    char str[250];
    int i = n - 1;
    printf("Noul imobil\n");
    ScanStruct(i);
}

```

```

void Edit()
{

    int i;
    do
    {
        printf("dati numarul elementului pe care doriti sa il editati : ");
        scanf("%d", &i);
        if (i > n)
        {
            printf("numarul nu trebuie sa depaseasca %d\n", n);
        }
    } while (i > n);
    i--;
    char str[255];
    printf("Editarea imobilului\n");
    printf("vechiul proprietar - %s\nnoul proprietar: ", imobile[i].proprietar);
    fflush(stdin);
    gets(str);
    imobile[i].proprietar = (char *)malloc((strlen(str) + 1) * sizeof(char));
    strcpy(imobile[i].proprietar, str);
    printf("vechiul tip - %s\nnoul tip: ", imobile[i].tip);
    fflush(stdin);
    gets(str);
    imobile[i].tip = (char *)malloc((strlen(str) + 1) * sizeof(char));
    strcpy(imobile[i].tip, str);
    printf("vechia adresa - %s\nnoua adresa: ", imobile[i].adresa);
    fflush(stdin);
    gets(str);
    imobile[i].adresa = (char *)malloc((strlen(str) + 1) * sizeof(char));
    strcpy(imobile[i].adresa, str);
    printf("vechia suprafata - %f\nnoua suprafata: ", imobile[i].adresa);
    scanf("%f", &imobile[i].suprafata);
    printf("vechiul cost - %f\nnoul cost: ", imobile[i].costul);
    scanf("%f", &imobile[i].costul);
    printf("%d", n);
}

int Search()
{
    printf("Ce pret va intereseaza : ");
    float price;
    scanf("%f", &price);
    float delta[n];
    for (int i = 0; i < n; i++)
    {
        if ((int)price == (int)imobile[i].costul)
        {
            return i;
        }
        delta[i] = abs(imobile[i].costul - price);
    }
    int ret = 0;

```

```

for (int i = 0; i < n - 1; i++)
{
    if (delta[i] <= delta[ret])
    {
        ret = i;
    }
}
return ret;
}
void Remove()
{
    int x;
    printf("nr elementului pe care doriti sa il eliminati");
    scanf("%d", &x);
    x--;
    for (int i = x; i < n - 1; i++)
        imobile[i] = imobile[i + 1];
    n--;
    imobile = (IB *)realloc(imobile, n * sizeof(IB));
    printf("eliminare efectuata cu succes");
}
void Free()
{
    for (int i = 0; i < n; i++)
    {
        free(imobile[i].proprietar);
        free(imobile[i].tip);
        free(imobile[i].adresa);
    }
    free(imobile);
    n = 0;
}
int main()
{
    int m;
    int s;
    do
    {
        Menu(&m);
        switch (m)
        {
            case 1:
                imobile = Creation();
                break;
            case 999:
                Demo();
                break;
            case 2:
                scanMatrix();
                break;
            case 3:

```



```

        printMatrix();
        break;
    case 4:
        Insert();
        break;
    case 5:
        Edit();
        break;
    case 6:
        s = Search();
        printf("cel mai apropiat pret de pretul cautat este al imobilului %d\n", s + 1);
        printf("detinut de %s si la pretul de %f", imobile[s].proprietar, imobile[s].costul);
        break;
    case 7:
        Order();
        break;
    case 8:
        Remove();
        break;
    case 9:
        Free();
        break;
    default:

        break;
    }
} while (m);
Free();
return 0;
}

```

Screenshoturi cu programul în acțiune:

```

1. Alocarea dinamica a memoriei pentru tabloul de structuri.
2. Introducerea elementelor tabloului de la tastatura. Sau 999 pentru Demo
3. Afisarea elementelor tabloului la ecran.
4. Adaugarea unui element nou la sfarsit.
5. Modificarea elementului tabloului.
6. Cautarea elementului tabloului.
7. Sortarea tabloului.
8. Eliminarea elementului indicat din tablou.
9. Eliberarea memoriei alocate pentru tablou
0. Iesire din program.
Optiunea - 1
Dati numarul de imobile : 3

```

Optiunea - 2

Citirea imobilelor

Imobilul 1

proprietar: Catalin

tip: castel

adresa: Harvard 123/2

suprafata : 5342

costul : 9876543210

Imobilul 2

proprietar: Lilian

tip: casa

adresa: str studentilor 21

suprafata : 75

costul : 300000

Imobilul 3

proprietar: Andrei

tip: Hotel

adresa: Bul. Moscovei 67

suprafata : 3000

costul : 9000000

Optiunea - 3

Nr.	Proprietar	tip	adresa	suprafata	costul
1	Catalin	castel	Harvard 123/2	5342.00 m^2	9876543488.00 \$
2	Lilian	casa	str studentilor 21	75.00 m^2	300000.00 \$
3	Andrei	Hotel	Bul. Moscovei 67	3000.00 m^2	9000000.00 \$

Optiunea - 4

Noul imobil

Imobilul 4

proprietar: Cost

tip: apartament

adresa: str florilor 23

suprafata : 22

costul : 30000

Optiunea - 3

Nr.	Proprietar	tip	adresa	suprafata	costul
1	Catalin	castel	Harvard 123/2	5342.00 m^2	9876543488.00 \$
2	Lilian	casa	str studentilor 21	75.00 m^2	300000.00 \$
3	Andrei	Hotel	Bul. Moscovei 67	3000.00 m^2	9000000.00 \$
4	Cost	apartament	str florilor 23	22.00 m^2	30000.00 \$

Optiunea - 5

dati numarul elementului pe care doriti sa il editati : 4

Editarea imobilului

vechiul proprietar - Cost

noul proprietar: Costea

vechiul tip - apartament

noul tip: apartament

vechia adresa - str florilor 23

noua adresa: str florilor 22

vechia suprafata - 0.000000

noua suprafata: 22

vechiul cost - 30000.000000

noul cost: 31124.51

Optiunea - 3

Nr.	Proprietar	tip	adresa	suprafata	costul
1	Catalin	castel	Harvard 123/2	5342.00 m^2	9876543488.00 \$
2	Lilian	casa	str studentilor 21	75.00 m^2	300000.00 \$
3	Andrei	Hotel	Bul. Moscovei 67	3000.00 m^2	9000000.00 \$
4	Costea	apartament	str florilor 22	22.00 m^2	31124.51 \$

Optiunea - 3							
Nr.	Proprietar	tip	adresa	suprafata	costul		
1	Catalin	hotel	eminescu 23	74.22 m^2	76366.41 \$		
2	Marius_01k	hotel	bul. moscovei 11	271.50 m^2	558204.00 \$		
3	daniel	palat	starda 3	214.00 m^2	356738.00 \$		
4	R Leta	palat	eminescu 23	242.83 m^2	1541263.13 \$		
5	Cristian	castel	eminescu 23	131.83 m^2	266567.00 \$		
6	Rita	hotel	botanica 23	192.38 m^2	347044.50 \$		
7	Dlinii	hotel	eminescu 23	246.67 m^2	196346.67 \$		
8	R Leta	apartament	bul. moscovei 11	136.22 m^2	1795272.63 \$		
9	Dlinii	spalatorie auto	livezilor 12	310.50 m^2	404581.50 \$		
10	Dlinii	casa	strada 2	102.83 m^2	433236.84 \$		
11	Ion B	spalatorie auto	botanica 23	31.83 m^2	60766.30 \$		

Optiunea - 6

Ce pret va intereseaza : 25000

cel mai apropiat pret de pretul cautat este al imobilului 1

detinut de Catalin_ si la pretul de 76366.414063

Optiunea - 7

1. sortare dupa pret descrescator

2. sortare dupa pret crescator

3.sortare dupa suprafata descrescatoare

orice alt numar. sortare dupa suprafata crescatoare1

Tabloul a fot sortat cu succes!

Optiunea - 3							
Nr.	Proprietar	tip	adresa	suprafata	costul		
1	R Leta	apartament	bul. moscovei 11	136.22 m^2	1795272.63 \$		
2	R Leta	palat	eminescu 23	242.83 m^2	1541263.13 \$		
3	Marius_01k	hotel	bul. moscovei 11	271.50 m^2	558204.00 \$		
4	Dlinii	casa	strada 2	102.83 m^2	433236.84 \$		
5	Dlinii	spalatorie auto	livezilor 12	310.50 m^2	404581.50 \$		
6	daniel	palat	starda 3	214.00 m^2	356738.00 \$		
7	Rita	hotel	botanica 23	192.38 m^2	347044.50 \$		
8	Cristian	castel	eminescu 23	131.83 m^2	266567.00 \$		
9	Dlinii	hotel	eminescu 23	246.67 m^2	196346.67 \$		
10	Catalin	hotel	eminescu 23	74.22 m^2	76366.41 \$		
11	Ion B	spalatorie auto	botanica 23	31.83 m^2	60766.30 \$		

Optiunea - 8

nr elementului pe care doriti sa il eliminati11

Optiunea - 8

nr elementului pe care doriti sa il eliminati5

Optiunea - 3							
Nr.	Proprietar	tip	adresa	suprafata	costul		
1	R Leta	apartament	bul. moscovei 11	136.22 m^2	1795272.63 \$		
2	R Leta	palat	eminescu 23	242.83 m^2	1541263.13 \$		
3	Marius_01k	hotel	bul. moscovei 11	271.50 m^2	558204.00 \$		
4	Dlinii	casa	strada 2	102.83 m^2	433236.84 \$		
5	daniel	palat	starda 3	214.00 m^2	356738.00 \$		
6	Rita	hotel	botanica 23	192.38 m^2	347044.50 \$		
7	Cristian	castel	eminescu 23	131.83 m^2	266567.00 \$		
8	Dlinii	hotel	eminescu 23	246.67 m^2	196346.67 \$		
9	Catalin	hotel	eminescu 23	74.22 m^2	76366.41 \$		

0. Iesire din program.

Optiunea - 3

Optiunea - 9

Matricea este goala

Demo

Optiunea - 1

Dati numarul de imobile : 25

Optiunea - 999

memoria utilizata de matrice 1101 B

Optiunea - 3						
Nr.	Proprietar	tip	adresa	suprafata	costul	
1	Maria	restaurant	studentilor 7/1	204.50 m ²	377740.72 \$	
2	daniel	spalatorie auto	starda 3	86.83 m ²	25429.76 \$	
3	R Sam	pestera	studentilor 7/1	285.67 m ²	49134.66 \$	
4	R Leta	casa	starda 3	214.00 m ²	316078.00 \$	
5	Ion B	palat	eminescu 23	241.50 m ²	4410031.50 \$	
6	R Leta	palat	livezilor 12	72.38 m ²	308968.88 \$	
7	R Leta	palat	starda 3	66.20 m ²	77696.73 \$	
8	Ion B	castel	botanica 23	256.38 m ²	1104719.88 \$	
9	Rita	castel	strada 2	300.00 m ²	990300.00 \$	
10	daniel	restaurant	studentilor 7/1	132.83 m ²	227809.16 \$	
11	Dlinii	pestera	strada 2	106.38 m ²	115948.75 \$	
12	ioneltuc	spalatorie auto	starda 3	79.00 m ²	139961.66 \$	
13	Dlinii	restaurant	botanica 23	19.20 m ²	25239.77 \$	
14	Dlinii	apartament	botanica 23	245.50 m ²	566368.50 \$	
15	R Leta	casa	starda 3	260.75 m ²	55800.50 \$	
16	daniel	castel	eminescu 23	269.75 m ²	226050.50 \$	
17	Catalin	restaurant	eminescu 23	212.00 m ²	476552.44 \$	
18	ioneltuc	gradina zoo	strada 1	268.57 m ²	632754.31 \$	
19	R Leta	casa	bul. moscovei 11	36.83 m ²	14766.82 \$	
20	Maria	casa	eminescu 23	266.75 m ²	389455.00 \$	
21	Ion B	casa	eminescu 23	81.22 m ²	159786.25 \$	
22	R Sam	hotel	eminescu 23	137.38 m ²	260737.75 \$	
23	Maria	casa	eminescu 23	136.57 m ²	681081.75 \$	
24	R Sam	spalatorie auto	strada 2	127.22 m ²	145835.81 \$	
25	Ion B	restaurant	eminescu 23	221.20 m ²	398534.34 \$	

Analiza datelor de ieșire:

- Aparent toate funcțiile lucrează corect și rezultatele afișate corespund așteptărilor (în cele mai multe cazuri uneori la căutarea după preț nu se găsește elementul dorit).
- Tabloul este sortat corect deși doar după cifre.
- În urma adăugării unui element sau eliminării nu apar erori.
- Indiferent dacă utilizatorul selectează eliberarea memoriei aceasta v-a fi eliberată la ieșirea din program.

Concluzii :

Acest program este cel mai complex program pe care l-am făcut la laboratoarele la PC, și deși nu îmi place să recunosc mi-a luat mai mult de 7 ore. Când operezi cu pointeri structuri și funcții pot apărea erori bizare și aparent imposibile. Structurile sunt niște containere de date destul de utile și practice. Am creat o structură care ocupă cel puțin 20 baiți și se poate mări dinamic în dependență de mărimea șirului de caractere care sunt înscrise.