Ministerul Educaţiei, Culturii și Cercetării
Universitatea Tehnică a Moldovei

Departamentul Ingineria Software și Automatică

RAPORT

Lucrarea de laborator nr. 2
la Structuri de date și algoritmi
Varianta 18

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

Chișinău – 2021

Lucrarea de laborator nr. 2

Tema : Fișiere

Scopul : Elaborarea unui program care să opereze cu fișiere.

Sarcina: Modificarea programuli precedent pentru a fi capabil să scrie structura de date într-un fișier apoi să fie capabil să citească această structură.

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

Rezumat succint la temă :

- Operarea cu fișiere ne eset asigurată de către biblioteca stdlib, care conține funcțiile necesare pentru deschiderea și închiderea fișierului.
- În primul rând am declarat fișierul **f** cu ajutorul cuvântului cheie FILE.
- În acest program am adăugat 3 funcții care sa permită operarea cu fișiere.
- Funcția FW() când este apelată încearcă să deschidă dișierul, dacă nu i se primește afițează asta la ecran, iar dacă deschide cu succes fișierul inscrie în el informațiile din matricea de structuri. Este important de menționat că funcția nu scrie pur și simplu matricea în format binar, deoarece structura respectivă conține pointeri informația cărora ar fi pierdută în caz contrar.
- Funcția FR() în principiu funcționează similar cu funcția FW() diferența fiind că citește matricea fișier, alocând memorie pentru ea, și pentru pinterii necesari.
- Funcția RM() șterge fișierul, în caz că utilizatorul are nevoie de această funcție.

Funcții pe care am învățat să le utilizez:

1. fopen - are două argumente, primul este denumirea fișierului iar al doilea este modul de deschidere, această funcție returnează un pointer de tip FILE.
2. putw / getw – este ce-a mai simplă metodă pe care am gasito pentru a scire și citi în fișier date de tip int.
3. fread / fwrite – poate scrie și citi în fișier orice fel de informație. Merită menționat că primul argument este un pointer iar din neatenție când operăm cu variabile care nu sunt pointeri putem face greșeala banală de a nu utiliza **&.**
4. fclose – închide fițierul, unele sisteme pot deschide un număr destul de mic de fișiere de aceea este bine de închis fișierele pe care nu le mai folosim.
5. Remove – șterge fișierul indicat.

Cod sursă in limbajul C :

```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 = 0;
```

```
const char numeFisier[15] = "lab.txt";

const char name[][30] = {"Catalin", "Marius", "Daniel", "Mirela", "Alex", "Colea", "Sandu", "Ion", "Maxim
o", "Melissa", "Petru", "Stas", "Vlad", "Crstian", "Ion", "Mihail", "Victor", "Vladislav", "Maria", "Vitalie", "Ni
coleta", "Sam"};
const char type[][30] = {"Apartament", "Birou", "Fabrica", "Magazin", "Mol", "Hotel", "Cladire Istorica", "T
eren gol", "Restaurant"};
const char addres[][30] = {"Strada Albisoara", "Strada Alexandru Bernardazzi", "Strada Alexandru cel B
un", "Strada Alexei Mateevici", "Strada Armeneasca", "Strada Bucuresti", "Strada Calea Iesilor", "Strad
a Mihail Kogalniceanu"};

FILE *f;

void Menu(int *m);
IB *Creation();
void Demo();
void ScanStruct(int i);
void scanMatrix();
void printMatrix();
void Order();
void Insert();
void Edit();
int Search();
void Remove();
void Free();

void FR()
{
    f = fopen(numeFisier, "rb");
    if (!f)
    {
        printf("nu s-a putut deschide acest fisier\n");
    }
    else
    {
        n = getw(f);
        printf("%d", n);
        imobile = (IB *)malloc(n * sizeof(IB));
        for (int i = 0; i < n; i++)
        {
            int tempSize;
            tempSize = getw(f);
            imobile[i].proprietar = (char *)malloc(tempSize);
            fread(imobile[i].proprietar, tempSize, 1, f);
            tempSize = getw(f);
            imobile[i].tip = (char *)malloc(tempSize);
            fread(imobile[i].tip, tempSize, 1, f);
            tempSize = getw(f);
            imobile[i].adresa = (char *)malloc(tempSize);
            fread(imobile[i].adresa, tempSize, 1, f);
            fread(&imobile[i].suprafata, sizeof(float), 1, f);
            fread(&imobile[i].costul, sizeof(float), 1, f);
        }
        printMatrix();
        printf("citirea s-a efectuat cu succes\n");
    }
```

```
    fclose(f);
}
void FW()
{
    f = fopen(numeFisier, "wb");
    if (!f)
    {
        printf("nu s-a putut deschide acest fisier\n");
    }
    else
    {
        if (n)
        {
            putw(n, f);
            for (int i = 0; i < n; i++)
            {
                putw(strlen(imobile[i].proprietar) + 1, f);
                fwrite(imobile[i].proprietar, strlen(imobile[i].proprietar) + 1, 1, f);
                putw(strlen(imobile[i].tip) + 1, f);
                fwrite(imobile[i].tip, strlen(imobile[i].tip) + 1, 1, f);
                putw(strlen(imobile[i].adresa) + 1, f);
                fwrite(imobile[i].adresa, strlen(imobile[i].adresa) + 1, 1, f);
                fwrite(&imobile[i].suprafata, sizeof(float), 1, f);
                fwrite(&imobile[i].costul, sizeof(float), 1, f);
            }
            printf("scrierea s-a efectuat cu succes\n");
        }
        else
            printf("matricea este goala nu avem ce scrie in fisier\nmai gandestete..\n");
    }
    fclose(f);
}

void RM()
{
    remove(numeFisier);
    printf("fisierul a fost sters cu succes\n");
}

int main()
{
    int m;
    do
    {
        Menu(&m);
        switch (m)
        {
        case 1:
            imobile = Creation();
            break;
        case 999:
            Demo();
            break;
        case 2:
            scanMatrix();
            break;
        case 22:
```

```
            FR();
            break;
        case 33:
            FW();
            break;
        case 3:
            printMatrix();
            break;
        case 4:
            Insert();
            break;
        case 5:
            Edit();
            break;
        case 6:
            Search();
            break;
        case 7:
            Order();
            break;
        case 8:
            Remove();
            break;
        case 666:
            RM();
            break;
        case 9:
            Free();
            break;
        default:

            break;
        }
    } while (m);
    Free();
    return 0;
}
void Menu(int *m)
{
    printf("\n22. Citirea din fisier.");
    printf("\n33. Scrierea in fisier.");
    printf("\n666. Sterge fisierul.\n");
    printf("\n1. Alocarea dinamica a memoriei pentru tabloul de structuri.");
    printf("\n2. Introducerea elementelor tabloului de la tastatura.\n999. pentru Demo ( umplerea matricii
cu elemente aleatorii )");
    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.");
    printf("\nOptiunea - ");
    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() % 22]);
        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, addres[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].supra
fata * (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);
    }
    if (totalMem >= 1024 * 1024)
        printf("\nmemoria utilizata de matrice %d MB\n", (int)totalMem / (1024 * 1024));
    else if (totalMem >= 1024)
        printf("\nmemoria utilizata de matrice %d KB\n", (int)totalMem / 1024);
    else
        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\ttip\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, imobile[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 suprafata d
escrescatoare\norice 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 fot sortat cu succes!\n");
}
void Insert()
{
    IB *tempImobile = realloc(imobile, ++n * sizeof(IB));
```

```
    if (tempImobile)
    {
        imobile = tempImobile;
    }
    else
    {
        printf("nu e posibil de inserat un element nou");
    }
    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 trebui 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;
        }
    }
    printf("cel mai apropiat pret de pretul cautat este al imobilului %d\n", ret + 1);
    printf("detinut de %s si la pretul de %f", imobile[ret].proprietar, imobile[ret].costul);
    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;
}
```

Screenshoturi cu programul în acțiune:
(insoțite de comentarii despre de operare al programului)

```
22. Citirea din fisier.
33. Scrierea in fisier.
666. Sterge fisierul.

1. Alocarea dinamica a memoriei pentru tabloul de structuri.
2. Introducerea elementelor tabloului de la tastatura.
999. pentru Demo ( umplerea matricii cu elemente aleatorii )
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 -
```

- când deschidem programul primul lucru pe care îl vedem este meniul principal cu cele 3 optiuni adaugate

80	Victor	Birou	Strada Albisoara	187.20 m^2	1335859.25 $
81	Sam	Birou	Strada Albisoara	39.57 m^2	58961.43 $
82	Alex	Birou	Strada Mihail Kogalniceanu	173.75 m^2	735310.00 $
83	Vlad	Magazin	Strada Bucuresti	232.75 m^2	336922.25 $
84	Maria	Fabrica	Strada Mihail Kogalniceanu	207.57 m^2	331906.72 $
85	Vladislav	Magazin	Strada Bucuresti	198.38 m^2	262549.31 $
86	Vlad	Birou	Strada Bucuresti	158.75 m^2	133826.25 $
87	Ion	Apartament	Strada Calea Iesilor	256.20 m^2	537763.81 $
88	Ion	Teren gol	Strada Bucuresti	156.00 m^2	203424.00 $
89	Stas	Fabrica	Strada Armeneasca	185.20 m^2	133529.20 $
90	Colea	Cladire Istorica	Strada Calea Iesilor	223.22 m^2	488410.22 $
91	Mihail	Cladire Istorica	Strada Armeneasca	132.00 m^2	52404.00 $
92	Maximo	Cladire Istorica	Strada Alexei Mateevici	163.75 m^2	201085.00 $
93	Mirela	Hotel	Strada Armeneasca	254.20 m^2	355625.78 $
94	Vladislav	Hotel	Strada Alexei Mateevici	147.83 m^2	241640.30 $
95	Mirela	Apartament	Strada Albisoara	284.83 m^2	237123.77 $
96	Vitalie	Apartament	Strada Alexandru Bernardazzi	111.38 m^2	75135.29 $
97	Victor	Cladire Istorica	Strada Calea Iesilor	290.83 m^2	547216.13 $
98	Nicoleta	Mol	Strada Alexei Mateevici	95.38 m^2	115785.25 $
99	Ion	Restaurant	Strada Alexandru cel Bun	61.50 m^2	128145.50 $
100	Melissa	Mol	Strada Mihail Kogalniceanu	78.20 m^2	87740.39 $

- am decis să utilizez comanda 1 și 999 pentru genera o matrice care conține 100 de imobile aceasta matrice ocupă 5KB de memorie

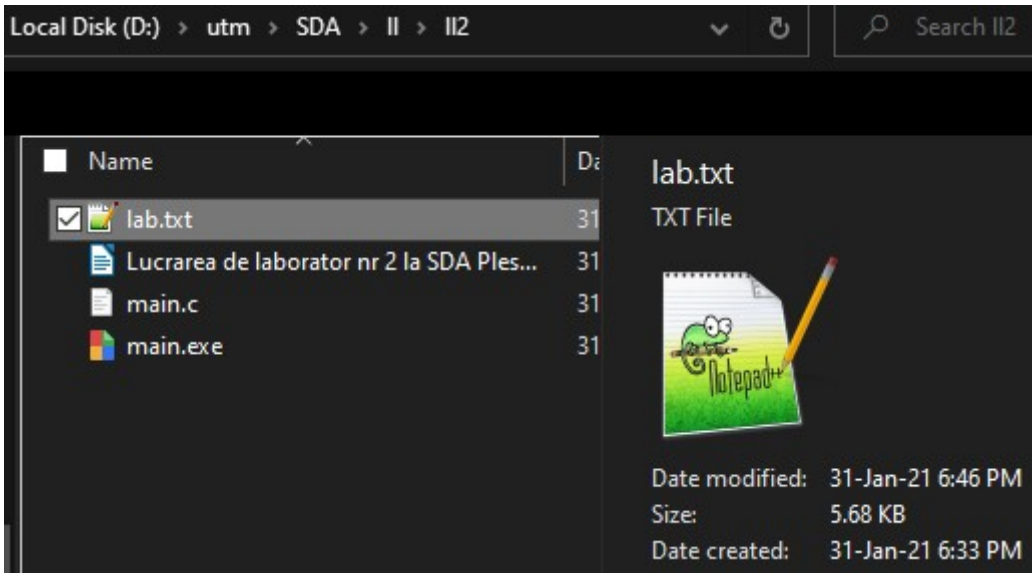
```
memoria utilizata de matrice 5 KB
```

Nr.	Proprietar	tip	adresa	suprafata	costul
1	Stas	Mol	Strada Calea Iesilor	309.00 m^2	4985715.00 $
2	Victor	Hotel	Strada Armeneasca	307.20 m^2	4973568.00 $
3	Victor	Fabrica	Strada Armeneasca	305.75 m^2	890344.00 $
4	Colea	Hotel	Strada Alexandru Bernardazzi	302.50 m^2	570817.50 $
5	Crstian	Teren gol	Strada Alexandru Bernardazzi	298.22 m^2	428346.50 $
6	Crstian	Restaurant	Strada Bucuresti	296.75 m^2	260744.33 $
7	Alex	Mol	Strada Calea Iesilor	294.50 m^2	238643.16 $
8	Victor	Cladire Istorica	Strada Calea Iesilor	290.83 m^2	547216.13 $
9	Mirela	Apartament	Strada Albisoara	284.83 m^2	237123.77 $
10	Ion	Magazin	Strada Alexandru cel Bun	276.50 m^2	682955.00 $
11	Petru	Teren gol	Strada Alexandru cel Bun	275.38 m^2	750396.88 $
12	Sam	Hotel	Strada Bucuresti	272.57 m^2	219231.31 $
13	Catalin	Apartament	Strada Mihail Kogalniceanu	267.83 m^2	212805.77 $
14	Ion	Teren gol	Strada Bucuresti	266.83 m^2	423149.16 $
15	Stas	Hotel	Strada Albisoara	263.83 m^2	80908.89 $
16	Daniel	Restaurant	Strada Alexei Mateevici	263.57 m^2	441934.03 $
17	Ion	Apartament	Strada Calea Iesilor	256.20 m^2	537763.81 $
18	Mirela	Hotel	Strada Armeneasca	254.20 m^2	355625.78 $
19	Crstian	Apartament	Strada Mihail Kogalniceanu	252.67 m^2	105362.00 $
20	Victor	Hotel	Strada Alexei Mateevici	249.20 m^2	1519870.75 $

- apoi am sortat matricea după suprafața descrescatoare (fără vre-un motiv anume)

```
Optiunea - 33
scrierea s-a efectuat cu succes
```

- am utilizat optiunea 33 pentru a scrie matricea într-un fișier



- în urma efectuării acestei operații am observat că în directoria curentă a apărut un fișier nou (lab.txt)

```
dNULNULNULENQNULNULNULStasNULEOTNULNULNULMolNULNAKNULNULNULStrada Calea IesilorN
NULNULNULTeren golNULGSNULNULNULStrada Alexandru BernardazziNULrFS•CP'ÑHBSNULNULN
NULNULNULTeren golNULEMNULNULNULStrada Alexandru cel BunNULNUL°‰CÎ37IEOTNULNULNUL
NULNULNULTeren golNULDC1NULNULNULStrada BucurestiNUL«j…C¥ÎHENQNULNULNULStasNULACK
NULNULNULTeren golNULCANNULNULNULStrada Alexei MateeviciNULNULÀuCÀé©G
NULNULNULVladislavNULVTNULNULNULRestaurantNULNAKNULNULNULStrada Calea IesilorNULN
NULNULNULTeren golNULNAKNULNULNULStrada Calea IesilorNULä8kCEOT•WHENQNULNULNULVla
NULNULNULTeren golNULCANNULNULNULStrada Alexei MateeviciNULUÕ_CêK½GACKNULNULNULCo
BSI
NULNULNULVladislavNULBSNULNULNULMagazinNULDC1NULNULNULStrada BucurestiNULNUL`FCª
NULNULNULTeren golNULDC1NULNULNULStrada BucurestiNULNULNULFSCNUL¨FHBSNULNULNULCa
NULNULNULVladislavNULACKNULNULNULHotelNULCANNULNULNULStrada Alexei MateeviciNULUÕ
ŠGBELNULNULNULMariusNULBSNULNULNULMagazinNULDC1NULNULNULStrada BucurestiNULNUL€á
NULNULNULVladislavNULBSNULNULNULMagazinNULGSNULNULNULStrada Alexandru Bernardazzi
NULNULNULTeren golNULDC1NULNULNULStrada BucurestiNUL«ª«Bã«LIBELNULNULNULMaximoNUL
NULNULNULTeren golNULESCNULNULNULStrada Mihail KogalniceanuNULNULNULGBhwEOTGENQNU
```

- în urma unei investigații am descoperit că în fișierul respectiv nu au fost înscrise doar caractere citibile (acest lucru m-a șocat) însă am decis că reiau testarea programului iar mai apoi să revin la acest subiect

```
Optiunea - 0
```

- am decis să închid programul pentru a mă asigura că matricea va fi citită din fișier

```
Optiunea - 22
100| Nr.|              Proprietar      |              tip      |      adresa    |      suprafata | costul |
|  1 |                 Stas |                  Mol |     Strada Calea Iesilor |      309.00 m^2 |  4985715.00 $ |
|  2 |               Victor |                Hotel |        Strada Armeneasca |      307.20 m^2 |  4973568.00 $ |
|  3 |               Victor |              Fabrica |        Strada Armeneasca |      305.75 m^2 |   890344.00 $ |
|  4 |                Colea |                Hotel |Strada Alexandru Bernardazzi |      302.50 m^2 |   570817.50 $ |
|  5 |              Crstian |           Teren gol |Strada Alexandru Bernardazzi |      298.22 m^2 |   428346.50 $ |
|  6 |              Crstian |           Restaurant |         Strada Bucuresti |      296.75 m^2 |   260744.33 $ |
|  7 |                 Alex |                  Mol |     Strada Calea Iesilor |      294.50 m^2 |   238643.16 $ |
|  8 |               Victor |     Cladire Istorica |     Strada Calea Iesilor |      290.83 m^2 |   547216.13 $ |
|  9 |               Mirela |           Apartament |         Strada Albisoara |      284.83 m^2 |   237123.77 $ |
| 10 |                  Ion |              Magazin | Strada Alexandru cel Bun |      276.50 m^2 |   682955.00 $ |
| 11 |                Petru |           Teren gol | Strada Alexandru cel Bun |      275.38 m^2 |   750396.88 $ |
| 12 |                 Sam |                Hotel |         Strada Bucuresti |      272.57 m^2 |   219231.31 $ |
| 13 |              Catalin |           Apartament |Strada Mihail Kogalniceanu |      267.83 m^2 |   212805.77 $ |
| 14 |                 Ion |           Teren gol |         Strada Bucuresti |      266.83 m^2 |   423149.16 $ |
| 15 |                Stas |                Hotel |         Strada Albisoara |      263.83 m^2 |   80908.89 $ |
| 16 |              Daniel |           Restaurant |  Strada Alexei Mateevici |      263.57 m^2 |   441934.03 $ |
| 17 |                 Ion |           Apartament |     Strada Calea Iesilor |      256.20 m^2 |   537763.81 $ |
| 18 |               Mirela |                Hotel |        Strada Armeneasca |      254.20 m^2 |   355625.78 $ |
| 19 |              Crstian |           Apartament |Strada Mihail Kogalniceanu |      252.67 m^2 |   105362.00 $ |
| 20 |               Victor |                Hotel |  Strada Alexei Mateevici |      249.20 m^2 |  1519870.75 $ |
```

- am executat programul din nou, opțiunea 22 a citit matricea din fișier cu succes cum spune mesajul afișat la ecran (pe care eu l-am progrmat să apară 😅)

```
| 95 |                Maria |              Fabrica |        Strada Armeneasca |      47.83 m^2 |    79738.16 $ |
| 96 |                Petru |                Hotel |        Strada Armeneasca |      42.50 m^2 |    53790.83 $ |
| 97 |                 Sam |                Birou |         Strada Albisoara |      39.57 m^2 |    58961.43 $ |
| 98 |                Colea |                  Mol |  Strada Alexei Mateevici |      29.22 m^2 |    41203.33 $ |
| 99 |                Vlad |           Restaurant |Strada Mihail Kogalniceanu |      27.50 m^2 |    53075.00 $ |
|100 |                Maria |           Restaurant |Strada Alexandru Bernardazzi |      19.67 m^2 |    17139.50 $ |
citirea s-a efectuat cu succes
```

```
| 81 |               Mirela |                Hotel | Strada Alexandru cel Bun |      61.00 m^2 |   260592.00 $ |
| 82 |               Marius |           Restaurant |Strada Mihail Kogalniceanu |      52.67 m^2 |    25150.73 $ |
| 83 |                Colea |           Teren gol |Strada Mihail Kogalniceanu |      49.75 m^2 |    33911.41 $ |
| 84 |                 Vlad |           Restaurant | Strada Alexandru cel Bun |      49.38 m^2 |    77871.43 $ |
| 85 |                Maria |              Fabrica |        Strada Armeneasca |      47.83 m^2 |    79738.16 $ |
| 86 |                Petru |                Hotel |        Strada Armeneasca |      42.50 m^2 |    53790.83 $ |
| 87 |                 Sam |                Birou |         Strada Albisoara |      39.57 m^2 |    58961.43 $ |
| 88 |                Colea |                  Mol |  Strada Alexei Mateevici |      29.22 m^2 |    41203.33 $ |
| 89 |                Vlad |           Restaurant |Strada Mihail Kogalniceanu |      27.50 m^2 |    53075.00 $ |
| 90 |                Maria |           Restaurant |Strada Alexandru Bernardazzi |      19.67 m^2 |    17139.50 $ |
```

- apoi am mai șters niște imobile pentru a mai verifica odată dacu scrierea și citirea funcționează

```
| 81 |               Mirela |                Hotel | Strada Alexandru cel Bun |      61.00 m^2 |   260592.00 $ |
| 82 |               Marius |           Restaurant |Strada Mihail Kogalniceanu |      52.67 m^2 |    25150.73 $ |
| 83 |                Colea |           Teren gol |Strada Mihail Kogalniceanu |      49.75 m^2 |    33911.41 $ |
| 84 |                 Vlad |           Restaurant | Strada Alexandru cel Bun |      49.38 m^2 |    77871.43 $ |
| 85 |                Maria |              Fabrica |        Strada Armeneasca |      47.83 m^2 |    79738.16 $ |
| 86 |                Petru |                Hotel |        Strada Armeneasca |      42.50 m^2 |    53790.83 $ |
| 87 |                 Sam |                Birou |         Strada Albisoara |      39.57 m^2 |    58961.43 $ |
| 88 |                Colea |                  Mol |  Strada Alexei Mateevici |      29.22 m^2 |    41203.33 $ |
| 89 |                Vlad |           Restaurant |Strada Mihail Kogalniceanu |      27.50 m^2 |    53075.00 $ |
| 90 |                Maria |           Restaurant |Strada Alexandru Bernardazzi |      19.67 m^2 |    17139.50 $ |
citirea s-a efectuat cu succes
```

- după încă o scriere și o citire acesta este rezultatul

Concluzii :

În urma efectuării acestui program am obținut abilitatea de a opera cu fișiere, ceea ce este extrem de important în orice porgram. Structura utilizată de mine nu a fost atât de bună fiind că nu îmi permite înscrierea ei pur și simplu, totuși înscrierea ei în fișier nu a fost dificilă doar că este nevoie de puțin mai mult cod. Dacă utilizatorul nu va vedea fișierul este mai bine de folosit fișiere binare fiind că am întâmpinat unele erori la citirea fișierelor în format simplu.