Abordarea problemei 2 la examen

1. Analiza enuntului

O firmă deține mai multe magazine alimentare. Firma, prin magazinele sale, comercializează un număr de **m** produse alimentare preambalate.

Să se scrie un program pentru a ține evidența produselor comercializate de firmă. În acest program sunt folosite informații referitoare la firmă care sunt stocate într-o structură și anume: numele firmei (printr-un pointer la caracter, numele firmei poate avea și spații), numărul de magazine și numărul de produse alimentare comercializate (doi întregi fără semn) și prețurile produselor (printr-un pointer la real în dublă precizie). Informațiile referitoare la stocurile de produse alimentare sunt stocate printr-un pointer la pointer la întreg fără semn, acest pointer NU este membru al structurii menționate mai sus.

Toate datele de intrare sunt citite dintr-un fișier care este dat ca argument al liniei de comandă.

În funcție de argumentele din linia de comandă se pot face următoarele prelucrări:

–m se afișează valoarea totală a mărfii deținută de firmă;

–a se ordonează și afișează matricea stocurilor alimentare în ordinea crescătoare a stocului total din fiecare magazin.

Opțiunea **–h** afișează pe monitor un mesaj de help (ce face programul și care este forma liniei de comandă). Prezența numelui fișierului este obligatorie în linia de comandă. Dacă numele fișierului nu este dat în linia de comandă atunci programul va afișa un mesaj de eroare și se va încheia.

Opțiunea implicită este –m.

Forma liniei de comandă: ./firma in.dat [-a][-m][-h]

Apelarea funcțiilor necesare pentru cele două prelucrări se va face prin intermediul pointerilor la funcții.

Structura firma
- Nume pointer la char
- n nr de magazine/linii
- m nr de produse/coloane

Ambele intregi fara semn
- Preturi pointer la real in dubla precizie (vector)

In main/ in afara structurii
- Stocuri pointer la pointer la intreg fara semn (matrice)

a, prin magazinele sale, comercializează un număr de **m** produse

nța produselor comercializate de firmă. În acest program sunt
te într-o structură și anume: numele firmei (printr-un pointer la
l de magazine și numărul de produse alimentare comercializate (doi
ointer la real în dublă precizie). Informațiile referitoare la stocurile
la pointer la întreg fără semn, acest pointer NU este membru al

Toate datele de intrare sunt citite dintr-un fișier care este dat ca argument al liniei de comandă.

În funcție de argumentele din linia de comandă se pot face următoarele prelucrări:

–m se afișează valoarea totală a mărfii deținută de firmă;

–a se ordonează și afișează matricea stocurilor alimentare în ordinea crescătoare a stocului total din fiecare magazin.

Opțiunea **–h** afișează pe monitor un mesaj de help (ce face programul și care este forma liniei de comandă).

Prezența numelui fișierului este obligatorie în linia de comandă. Dacă numele fișierului nu este dat în linia de comandă atunci programul va afișa un mesaj de eroare și se va încheia.

Opțiunea implicită este –m.

Forma liniei de comandă: ./firma in.dat [-a][-m][-h]

Apelarea funcțiilor necesare pentru cele două prelucrări se va face prin intermediul pointerilor la funcții.

Numele proiectului

2. Analiza baremului (pentru nota 5)

A1. Funcție de citire a informațiilor referitoare la firma (funcția are ca parametru fișierul din care se citesc datele – fișierul va fi deschis în main-) și returnează un pointer la structura cu informațiile despre firmă.	
A1.1. Scrierea funcției de citire	0.25
A1.2. Citirea numelui firmei (care poate conține spații)	0.25
A1.3. Citirea numărului de magazine și a numărului de produse	0.25
A1.4. Citirea prețurilor produselor alimentare (funcție care returmează pointer) și apelul corect al funcției	0.25
A1.5 Alocare pentru pointer la caracter și pointer la real – cu spațiul strict necesar	0.25
A1.6 Alocare pentru pointer la structură cu spațiul strict necesar	0.25
A1.7. Apelul corect al funcției de citire (a tuturor informațiilor) din main	0.25
A1.8. Funcționarea corectă a funcției la apelul din main	0.25
A2. **Funcție** de citire a stocurilor de alimente (funcția primește ca parametri fișierul din care se citesc datele, un pointer la strucură și returnează un pointer la pointer)	0
A2.1. Scrierea funcției de citire a stocurilor de alimente	0.25
A2.2. Alocare pentru pointer la pointer (cu spațiul strict necesar)	0.25
A2.3. Apelul corect al funcției de citire din main	0.25
A2.4. Funcționarea corectă a funcției la apelul din main	0.25
A3. **Funcție** de afișare a informațiilor despre firmă pe monitor	0
A3.1. Scrierea funcției de afișare	0.25
A3.2. Apelul corect al funcției din main	0.25
A3.3. Funcționarea corectă a funcției la apelul din main	0.25
A4. Construirea corectă a proiectului; definirea corectă a variabilelor cu care se lucrează	0.25
A5. Deschiderea fișierului, verificarea deschiderii și închiderea fișierului cu care s-a lucrat	0.25

2. Analiza baremului (pentru nota 5)

A6. Funcție pentru calculul valorii totale a mărfii, apelul corect din main (prin pointer la funcții) și funcționarea corectă (funcția are ca parametru un pointer la structură și un pointer la pointer)	1.25
A7. Funcție pentru ordonarea matricei, apelul corect din main (prin pointer la funcții) și funcționarea corectă (funcția are ca parametru un pointer la structură și un pointer la pointer)	1.5
A8. Eliberarea zonelor de memorie alocate	0.25
A9. Analiza liniei de comandă	0
A9.1 Identificarea fișierului cu care se lucrează	0.5
A9.2 Identificarea opțiunilor de prelucrare	0.5
A9.3 Tratarea inexistenței fișierului și afișarea mesajului de help	0.5
A10. Folosirea pointerilor la funcții pentru apelarea funcțiilor de prelucrare	0.75
A11. Funcție pentru afișarea rezultatelor prelucrărilor, apelul corect din main și funcționarea corectă la apelul din main	0.5

2. Analiza baremului (pentru nota 5)

- A1 2.00 p +
- A2 1.00 p
- A3 0.75 p
- A4 0.25 p
- A5 0.25 p
- A6 1.25 p
- A8 0.25 p

_____

5.75 p

A1 si A2 Alocarile si dealocare2d

```c
void* xmalloc(size_t nrOcteti)
{
    void *p=0;
    p=malloc(nrOcteti);
    if(!p)
    {
        fprintf(stderr,"Memorie insuficienta");
        exit(EXIT_FAILURE);
    }
    return p;
}

unsigned int** aloca2d(size_t n, size_t m)
{
    unsigned int **a;
    int i;
    a=(unsigned int**)xmalloc(n*sizeof(unsigned int*));
    for(i=0;i<n;i++)
        a[i]=(unsigned int*)xmalloc(m*sizeof(unsigned int));
    return a;
}
```

```c
void dealoca2d(unsigned int** a, size_t n)
{
    int i;
    for(i=0;i<n;i++)
    {
        free(a[i]);
        a[i]=0;
    }
    free(a);
}
```

A1 si A2 citirea datelor (3p)

```
unsigned int** citireM(FILE *f, FIRMA *fir)
{
    unsigned int **a;
    int i,j;
    a=aloca2d(fir->n,fir->m);
    for(i=0;i<fir->n;i++)
        for(j=0;j<fir->m;j++)
        {
            fscanf(f,"%u",&a[i][j]);
        }
    return a;
}
```

```
double* citireV(unsigned int m, FILE *f)
                {
    int i;
    double *v;
    v=(double*)xmalloc(m*sizeof(double));
    for(i=0;i<m;i++)
        fscanf(f,"%lf",&v[i]);
    return v;
                }
```

```
FIRMA* citireInformatii(FILE *f)
                {
    FIRMA *fir;
    char aux[100];

    fir =(FIRMA*)xmalloc(sizeof(FIRMA));

    fgets(aux,99,f);
    aux[strlen(aux)-1]='\0';
    fir->denumire=(char*)xmalloc(strlen(aux)*sizeof(char));
    strcpy(fir->denumire, aux);

    fscanf(f,"%u %u",&fir->n,&fir->m);

    fir->pret=citireV(fir->m, f);

    return fir;

                }
```

Sirul de caractere il citesc cu fgets pentru a permite spatii, scanf citeste pana la spatiu!

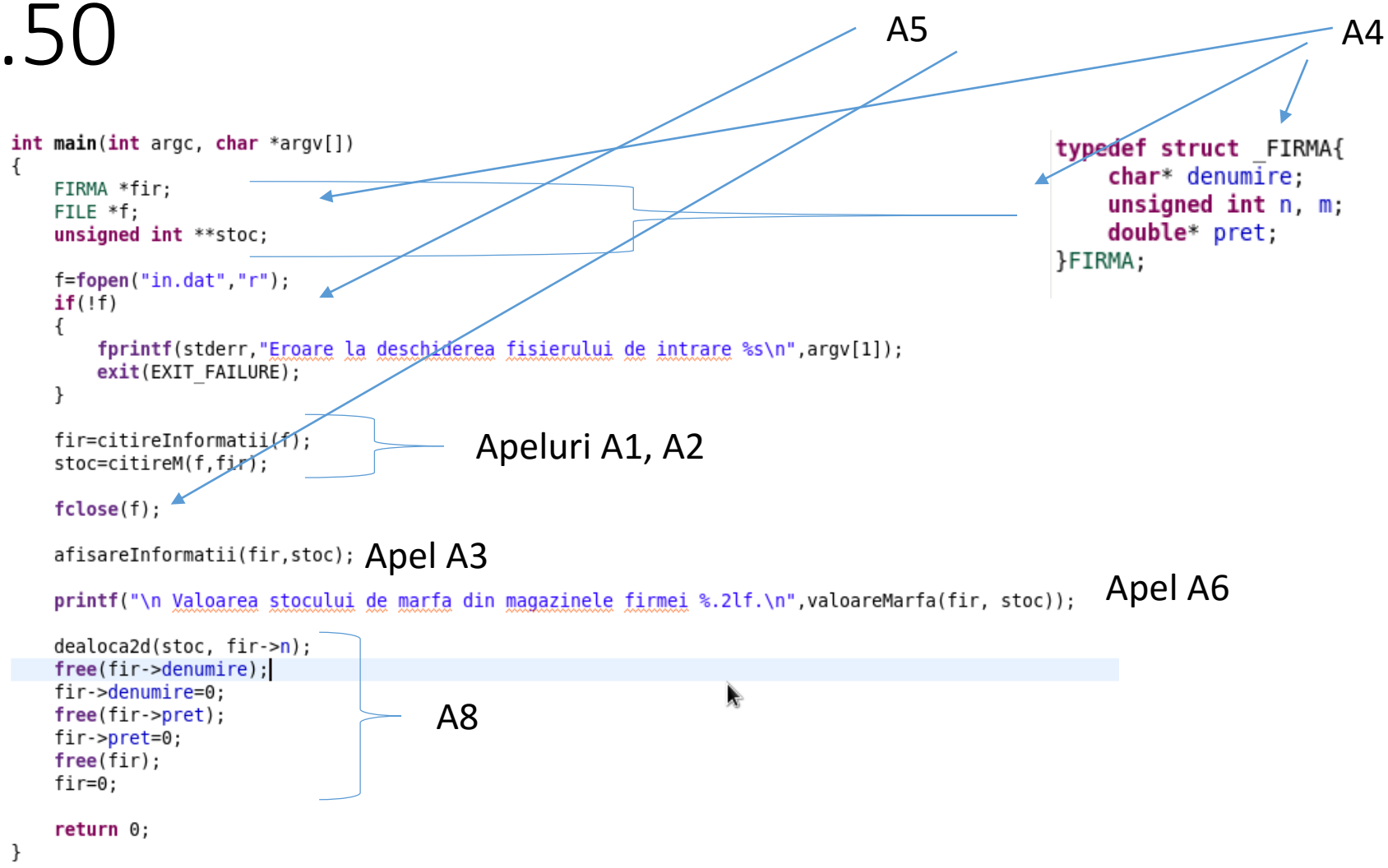
A3 Afisarea (0.75p)

```c
void afisareM(unsigned int **a, unsigned int n, unsigned int m)
{
    int i,j;
    for(i=0;i<n;i++)
    {
        printf("\t");
        for(j=0;j<m;j++)
            printf("%4u ",a[i][j]);
        printf("\n");
    }
}
```

```c
void afisareV(double *v, unsigned int m)
{
    int i;
    for(i=0;i<m;i++)
        printf("%.4lf ", v[i]);
}
```

```c
void afisareInformatii(FIRMA *fir, unsigned int** stoc)
{
    printf("\n\t Firma \"%s\" detine %u magazine in cadrul carora comercializeaza %u tipuri de produse.", fir->denumire, fir->n, fir->m);
    printf("\n Preturile sortimentelor comercializate sunt:\n\t");
    afisareV(fir->pret, fir->m);
    printf("\n Stocurile din magazinele firmei sunt:\n");
    afisareM(stoc, fir->n, fir->m);
}
```

Apelurile functiilor implementate pentru a obtine 5.50

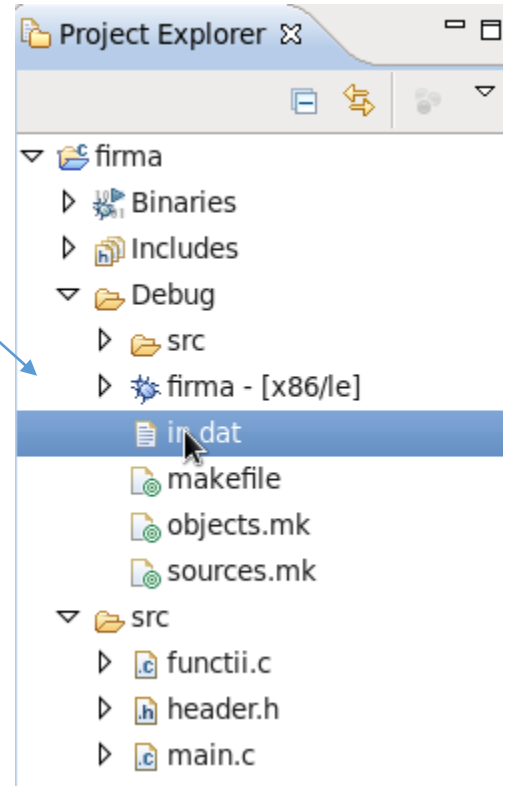


header.h | functii.c | main.c | in.dat

```
Padurea cu alune
3 5
12.5 7.49 8.56 3.277 78.2314
15 14 13 12 11
10 9 8 7 6
5 4 3 2 1
```

A4

```
typedef struct _FIRMA{
    char* denumire;
    unsigned int n, m;
    double* pret;
}FIRMA;
```




```
File  Edit  View  Search  Terminal  Help
[Laborator@localhost ~]$ cd workspace/firma/Debug/
[Laborator@localhost Debug]$ ./firma

        Firma "Padurea cu alune" detine 3 magazine in cadrul carora comercializ
eaza 5 tipuri de produse.
 Preturile sortimentelor comercializate sunt:
       12.5000 7.4900 8.5600 3.2770 78.2314
 Stocurile din magazinele firmei sunt:
         15   14   13   12   11
         10    9    8    7    6
          5    4    3    2    1

 Valoarea stocului de marfa din magazinele firmei 2259.65.
[Laborator@localhost Debug]$
```

5.75 + 1.5 = 7.25 pentru A7

```
    printf("\n Valoarea stocului de marfa din magazinele firmei %.2lf.\n",valoareMarfa(fir, stoc));

    ordonare(fir, stoc);
    printf("\n Dupa ordonare:\n");
    afisareM(stoc,fir->n, fir->m);

    dealoca2d(stoc, fir->n);
    free(fir->denumire);
    fir->denumire=0;
```

```
Laborator@localhost Debug]$ ./firma

        Firma "Padurea cu alune" detine 3 magazine in cadrul carora comercializeaza 5 tipuri de produse.
Preturile sortimentelor comercializate sunt:
       12.5000 7.4900 8.5600 3.2770 78.2314
Stocurile din magazinele firmei sunt:
         15   14   13   12   11
         10    9    8    7    6
          5    4    3    2    1

Valoarea stocului de marfa din magazinele firmei 2259.65.

Dupa ordonare:
          5    4    3    2    1
         10    9    8    7    6
         15   14   13   12   11
```

7.25+1=8.25 pentru A9.1 si A9.3

- A9.1

Forma liniei de comanda este:

./firma in.dat …

argv[0]	argv[1]	…	argv[argc-1]
firma	In.dat		

Daca argc =1, in linia de comanda am doar firma in argv[0]

Daca argc=2, in linia de comanda am firma in argv[0] si posibil numele fisierului in argv[1]. Daca primul character din argv[1] este diferit de – atunci este un posibil fisier. Voi testa argv[1][0]!='-'.

A9.1 (pentr 0.5p)

```
int fflag = 1;
if(argc==1)
    fflag = 0;
else
    if(argv[1][0]=='-')
        fflag = 0;
```

```
int fflag = 1;
if(argc==1)
    fflag = 0;
else
    if(argv[1][0]=='-')
        fflag = 0;

if(fflag)
{
    f=fopen(argv[1],"r");
    if(!f)
    {
        fprintf(stderr,"Eroare la deschiderea fisierului de intrare %s\n",argv[1]);
        exit(EXIT_FAILURE);
    }

    fir=citireInformatii(f);
    stoc=citireM(f,fir);

    fclose(f);

    afisareInformatii(fir,stoc);

    printf("\n Valoarea stocului de marfa din magazinele firmei %.2lf.\n",valoareMarfa(fir, stoc));

    ordonare(fir, stoc);
    printf("\n Dupa ordonare:\n");
    afisareM(stoc,fir->n, fir->m);

    dealoca2d(stoc, fir->n);
    free(fir->denumire);
    fir->denumire=0;
    free(fir->pret);
    fir->pret=0;
    free(fir);
    fir=0;
}

return 0;
```

A9.3 (0.5p)

```
if(fflag)
{
    f=fopen(argv[1],"r");
    if(!f)
    {
        fprintf(stderr,"Eroare la deschiderea fisierului de intrare %s\n",argv[1]);
        exit(EXIT_FAILURE);
    }

    fir=citireInformatii(f);
    stoc=citireM(f,fir);

    fclose(f);

    afisareInformatii(fir,stoc);

    printf("\n Valoarea stocului de marfa din magazinele firmei %.2lf.\n",valoareMarfa(fir, stoc));

    ordonare(fir, stoc);
    printf("\n Dupa ordonare:\n");
    afisareM(stoc,fir->n, fir->m);

    dealoca2d(stoc, fir->n);
    free(fir->denumire);
    fir->denumire=0;
    free(fir->pret);
    fir->pret=0;
    free(fir);
    fir=0;
}
else
{
    printf("\n Fisier inexistent in linia de comanda!");
    help();
}
```

A10. inca 0.75p => nota 9 Pointeri la fctii

- Am de apelat functiile de prelucrare:

double valoareMarfa(FIRMA *fir, unsigned int stoc);**

void ordonare(FIRMA *fir, unsigned int stoc);**

- Vom folosi:
- **typedef double (*pf1)(FIRMA*, unsigned int**);**
- **typedef void (*pf2)(FIRMA*, unsigned int**);**
- Am creat doua sinonime pf1 si pf2 pentru cei doi pointeri la fctiile de prelucrare.

A10. inca 0.75p => nota 9 Pointeri la fctii si 0.25p din A

- In main:

pf1 p1=&valoareMarfa;

pf2 p2=&ordonare;

- Apelurile vor fi:

printf("\n Valoarea stocului de marfa din magazinele firmei %.2lf.\n",**p1(fir, stoc)**);

p2(fir, stoc);

printf("\n Dupa ordonare:\n");

afisareM(stoc,fir->n, fir->m);

9.5 daca rezolvam si A9.2

Procesare optiuni in linia de comanda

Definim flag-uri

int iflag = 0, mflag = 0, aflag = 0, hflag = 0;

iflag = invalid cmd line = 0 valida si 1 nu e valida

m/a/hflag = 1 activ optiunea m/a/h

Ramane sa parcurgem lista de argumente si sa setam flagurile

A9.2 Procesare linie de comanda

```
switch(argc)
{
case 1:
    fflag = 0; break;
case 2:
    if(argv[1][0]=='-')
        fflag = 0;
    else
        mflag = 1;
    break;
default:
    if(argv[1][0]=='-')
        fflag=0;
    else
    {
        for(i=2;i<argc;i++)
        {
            if(argv[i][0]=='-')
                switch(argv[i][1])
                {
                case 'm':
                    mflag = 1;
                    break;
                case 'a':
                    aflag = 1;
                    break;
                case 'h':
                    hflag = 1;
                    break;
                default:
                    iflag = 1;
                    break;
                }
            else
                iflag = 1;
        }
    }
    break;
}
```

```
int fflag = 1;
if(argc==1)
    fflag = 0;
else
    if(argv[1][0]=='-')
        fflag = 0;
```

Includem si verificarea existentei fisierului.
Daca avem fisier citim si afisam datele apoi verificam validitatea liniei de comanda si aplicam procesarile necesare.

A9.2 Procesare linie de comanda

```
if(fflag)
{
    f=fopen(argv[1],"r");
    if(!f)
    {
        fprintf(stderr,"Eroare la deschiderea fisierului de intrare %s\n",argv[1]);
        exit(EXIT_FAILURE);
    }

    fir=citireInformatii(f);
    stoc=citireM(f,fir);

    fclose(f);

    afisareInformatii(fir,stoc);

    if(iflag==0)
    {
        rezultatePrelucrari(fir, stoc, aflag, mflag, hflag);
    }
    else
    {
        printf("\n Optiune invalida.\n");
        help();
    }
    dealoca2d(stoc, fir->n);
    free(fir->denumire);
    fir->denumire=0;
    free(fir->pret);
    fir->pret=0;
    free(fir);
    fir=0;
}
else
{
    printf("\n Lipseste fisierul de intrare din linia de comanda.\n");
    help();
}
```

A10 cele 0.5p pentru 10

Functia nu returneaza nimic, primeste datele de intrare necesare la apelul prelucrarilor si flagurile pentru a selecta optiunea de prelucrare.
Functia va permite optiuni multiple in varianta din dreapta, dar poate fi implementata si pentru optiune unica cu un switch case.

```
void rezultatePrelucrari(FIRMA *fir, unsigned int** stoc, int aflag, int mflag, int hflag)
{
    pf1 p1=&valoareMarfa;
    pf2 p2=&ordonare;
    double valMarfa;

    if(mflag)
    {
        printf("\n Optiunea m");
        valMarfa=p1(fir,stoc);
        printf("\n Valoarea stocului de marfa din magazinele firmei este: %.4lf.\n",valMarfa);
    }
    if(aflag)
    {
        printf("\n Optiunea a");
        p2(fir, stoc);
        printf("\n Dupa ordonarea crescatoare in functie de stocul total din fiecare magazin:\n");
        afisareM(stoc,fir->n, fir->m);
        printf("\n");
    }
    if(hflag)
    {
        printf("\n Optiunea h");
        help();
    }
}
```