Facultatea de Inginerie Electrică

UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

EUROPEAN UNIVERSITY OF TECHNOLOGY

Facultatea de Inginerie Electrica

PCLP 2

Programarea calculatoarelor si limbaje de programare 2

PCLP2

An I semestrul II

"Coding is easy when you C it in action."

Cap. 6

Fisiere in C

6. 1 Definire si declarare fisiere

- ❑ Stream-uri si fisiere
- ❑ Clasificare stream-uri si fisiere.

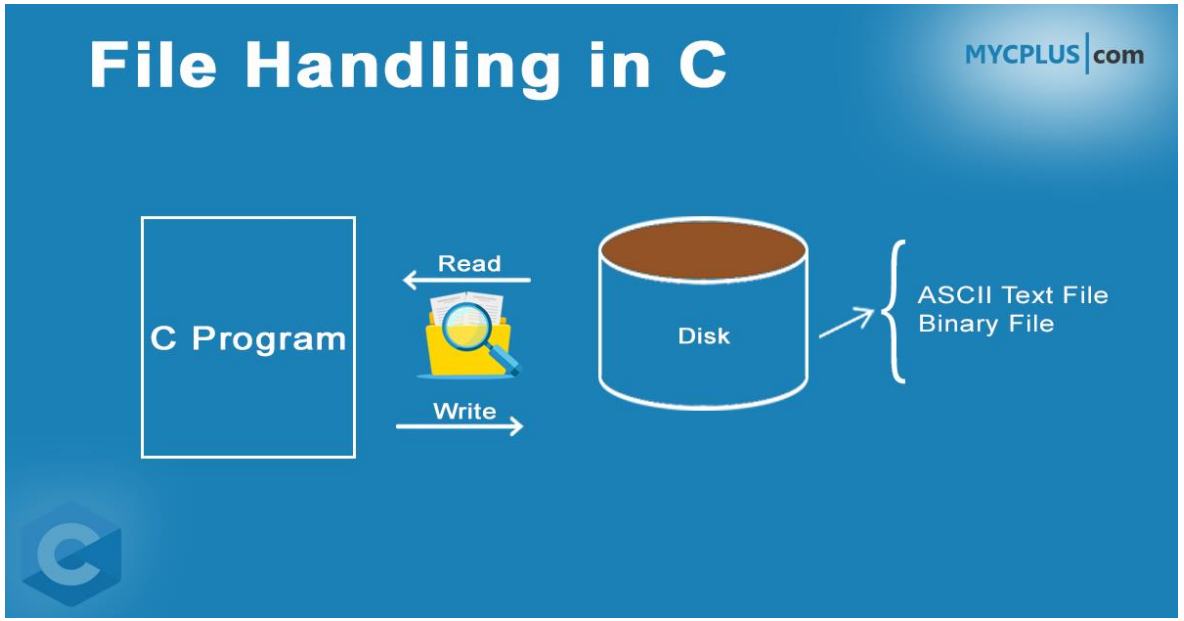
6.2 Moduri de tratare a fisierelor

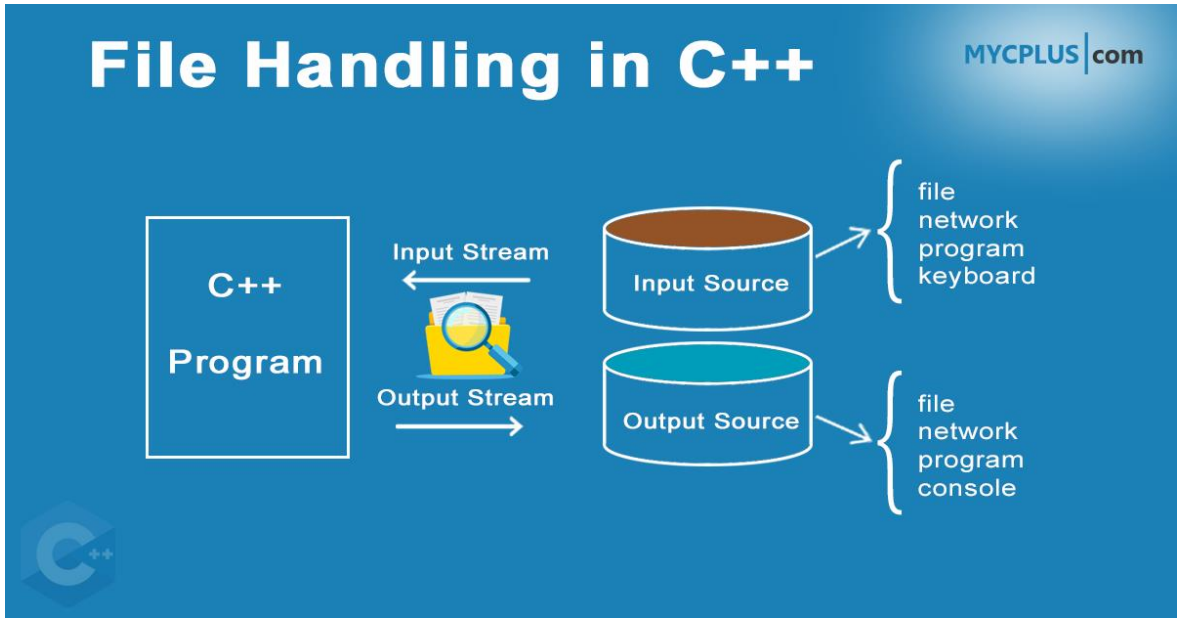
6.3 Functii de nivel inferior

6.4 Functii de nivel superior

6.1 Definire si declarare fisiere in C

Fisiere in C , fisiere si stream-uri in C++





- Compilatoarele C se bazează pe biblioteci externe pentru a efectua operațiuni de intrare sau de ieșire, Ex.<stdio.h> care oferă funcții de I/O printf() și fscanf(), fprintf().

- C++ utilizeaza bibliotecile <iostream> și <fstream>, iar fișierele sunt prelucrate prin stream-uri adica fluxuri de date în/din programe.

6.1 Definire si declarare fisiere in C

Stream-uri si fisiere

DEFINIRE

Stream (flux)= forma abstracta, independenta de tipul echipamentelor utilizate (terminale, drivere de disc, drivere de unitati fizice, etc) care realizeaza legatura dintre programul utilizatorului si fisierele de I/O utilizate de acesta.
Rol: Prin intermediul lor se realizeaza operatiile de citire/scriere din/in fisiere.

Fisier = colectie ordonata de inregistrari (articole) aflata pe diferite suporturi (magnetic, optic, etc) in care se stocheaza datele citite/scrise in programul utilizator. Fisierele in C/C++ au forma unei **structuri** numita **FILE**, care contine informatii despre un anumit stream .

Rol: Mediul de programare (CodeBlocks, Visual C++,etc) contine rutinele de I/O destinate manipularii structurii FILE, insa pentru utilizarea ei este necesara declararea unui pointer la aceasta structura. **Majoritatea functiilor specifice fisierelor in C sunt continute in <stdio.h>** si au un “f” adaugat in fata numelui . Ex. fscanf(), fprintf(), etc.

6.1 Definire si declarare fisiere in C

Stream-uri si fisiere

DEFINIRE

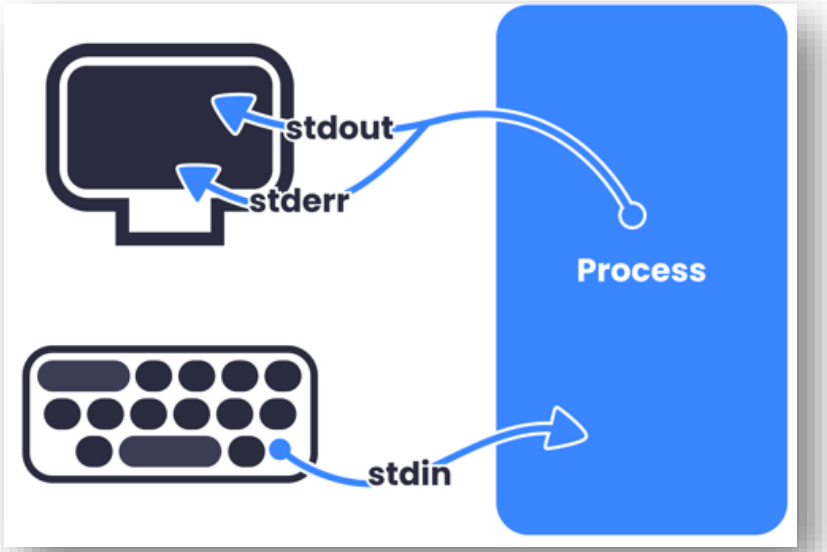
- **Stream text** = secventa de caractere ASCII, organizata pe linii, terminata optional cu caracterul linie noua. Ex.: .txt, .c, .cpp, etc.
- **Stream binar** = secventa ordonata de caractere (octeti).Ex. fis. executabile Ex.: exe, .bat,.com, etc.

DEFINIRE

La lansarea in executie a unui program C/C++ se deschid automat urmatoarele stream-uri:

- **stdin:** stream de intrare (intrare standard) de tip text. Ex. tastatura
- **stdout:** stream de iesire (iesire standard) de tip text. Ex.monitorul
- **stderr:** stream de iesire erori (iesire standard erori) de tip text.

La inchiderea programului C/C++ stream-urile sunt automat inchise.



6.1 Definire si declarare fisiere in C

Stream-uri si fisiere

DEFINIRE

Fisier: in C poate fi orice fisier text / binar de pe HDD, DVD, flash memory, carduri de memorie, etc.

Accesul la informatia continuta in fisiere: se obtine prin

- **deschiderea** fisierului respectiv
- **operatii**: citirea/scrierea / informatii, etc.
- **inchiderea** fisierului

Indicator de pozitie: la deschiderea fisierului

- se pozitioneaza pe inceputul de fisier, apoi
- se incrementeaza/decrementeaza pe parcursul parcurgerii fisierului,
- in final este pozitionat pe sfarsitul de fisier (eof)

6. 2. Moduri de tratare a fisierelor in C

La nivel inferior (low level)

Caracteristici:

- se utilizeaza mai rar pentru ca depind de SO, functiile utilizate la acest nivel sunt incluse in <io.h>, <stat.h>, <fcntl.h>

Exemple functii :

- **open, creat, read, write, lseek, close**

La nivel superior (high level)

Caracteristici:

- se utilizeaza mai frecvent, functiile sunt incluse in <stdio.h> pentru C si <fstream> pentru C++

Exemple functii :

- **fopen, fclose,fputc, fgetc,fseek, fprintf, fscanf,feof,ferror, rewind, remove, fflush**

6.3 Functii la nivel inferior

Deschidere fisier: open()

SINTAXA

int **open**(const char *path, int access [,unsigned mode]);

unde path = numele fisierului inclusiv calea spre fisier intre ghilimele “”

access = variabila de tip intreg care indica modul de deschidere a fisierului

Returneaza -1 pentru eroare (sau valori negative).Daca deschiderea fisierului se realizeaza cu **success functia returneaza valoare pozitiva**

access	Semnificatie
O_RDONLY	Citire
O_WRONLY	Scriere
O_RDWR	Citire/Scriere
O_APPEND	Adaugare
O_CREAT	Daca fisierul exista nu are efect. Daca nu exista este creat
O_TRUNC	Daca fisierul exista este trunchiat la 0
O_BINARY	Deschiderea fisierului in mod binar
O_TEXT	Deschiderea fisierului in mod text

mode	Semnificatie
S_IWRITE	Permisiune de scriere
S_IREAD	Permisiune de citire
S_IREAD|S_IWRITE	Permisiune de citire si scriere

6.3 Functii la nivel inferior

Inchidere fisier: close()

SINTAXA

int **close**(int handle);

Efect: Inchide fisierul asociat cu handle.

Returneaza -1 pentru eroare si 0 daca operatia s-a efectuat cu success.

6.3 Functii la nivel inferior

Exemple

EXEMPLU

Ex: Deschiderea unui fisier text.txt si testare daca fisierul e accesibil pentru citire si scriere

```
#include <stdio.h>   //pentru printf
#include <fcntl.h>   //pentru  O_RDWR
#include <io.h>      //pentru open,close
int main(void)
{int test;
if ((test=open("text.txt",O_RDONLY|O_RDWR))==-1)
            {printf("eroare la deschidere fisier!\n");}
else //operatii in fisier…
….
close(test); return 0;}
```

6.3 Functii la nivel inferior

Creare fisier: creat()

SINTAXA

int **creat**(const char *path, int amode);

unde path = calea spre noul fisier ce va fi creat
amode = variabila de tip intreg care indica modul de creare a fisierului

Returneaza -1 pentru eroare, sau valori negative .Daca creearea se realizeaza **cu success functia returneaza valoare pozitiva**

amode	Semnificatie
S_IWRITE	Permisiune de scriere
S_IREAD	Permisiune de citire
S_IREAD|S_IWRITE	Permisiune de citire si scriere

6.3 Functii la nivel inferior

Citire din fisier: read()

SINTAXA

int **read**(int handle, void *buf, unsigned len);

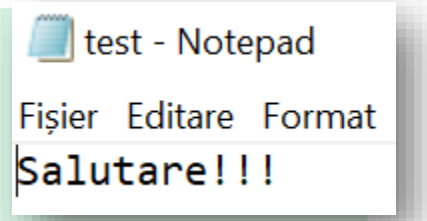
unde len = nr de octeti ce se incearca a fi cititi in bufferul de memorie buf din fisierul asociat cu handle la deschiderea cu functia open

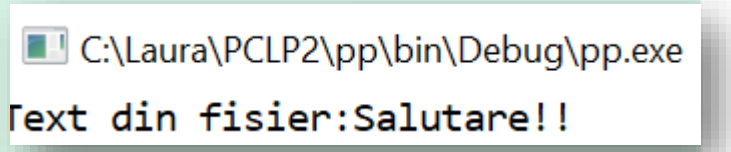
Returneaza val <0 =eroare, 0 =end-of-file,sau nr de octeti cititi cu succes

EXEMPLU Daca fisierul de intrare text.txt nu exista, programul nu afiseaza nimic

Ex: Citirea unui sir de maxim 15 caractere din fisierul text.txt si afisarea lui pe ecran

```
#include <stdio.h>   //pentru printf
#include <string.h> //pentru strlen
#include <fcntl.h>   //pentru  O_RDWR
#include <io.h>      //pentru open, write si close
int main(void)
{int test; char msg[15]; //msg buffer memorie fisier asociat
if ((test=open("test.txt",O_RDONLY|O_RDWR))==-1)   {printf("eroare la deschidere fisier!\n");}
else read(test,msg,10);
printf("Text din fisier:%s",msg);close(test);return 0;}
```





6.3 Functii la nivel inferior

Scriere in fisier: write()

SINTAXA

int **write**(int handle, void *buf, unsigned len);

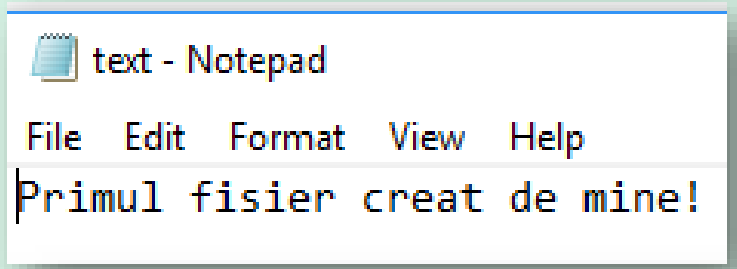
unde len = nr de octeti ce se incearca a fi scrisi din bufferul de memorie buf in fisierul asociat handle

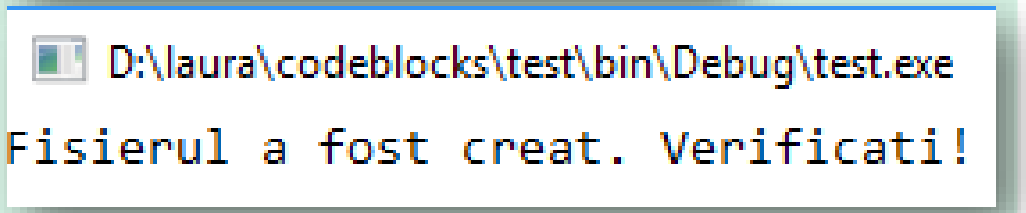
Returneaza valoare negativa pentru eroare, 0 pentru end-of-file, sau nr de octeti scrisi cu succes

EXEMPLU Daca fisierul de iesire text.txt nu exista programul il creeaza si scrie in el textul

Ex: Crearea, deschiderea unui fisier numit “text.txt” si scrierea in fisier a unui text

```
#include <stdio.h>   //pentru printf
#include <string.h> //pentru strlen
#include <fcntl.h>   //pentru O_CREAT si O_RDWR
#include <io.h>      //pentru open, write si close
int main(void)
{int test; char msg[]="Primul fisier creat de mine!";
if ((test=open("text.txt",O_CREAT|O_RDWR))==-1)
   {printf("eroare la deschidere fisier!\n");}
 else {printf(“Fisierul a fost creat. Verificati!\n");}
 write(test,msg,strlen(msg));close(test); return 0;}
```





6.4 Functii la nivel superior

Functii pentru fisiere in C (<stdio.h>)

	Functie	Semnificatie
Functii deschidere/ inchidere fisiere	fopen()	deschide un fisier
	fclose()	inchide un fisier
Functii scriere in fisiere	fputc(),putc()	scrie un caracter intr-un fisier
	fputs()	scrie un sir de caractere intr-un fisier
	fprintf()	scrie date formatate intr-un fisier
Functii citire din fisiere	fgetc(),getc()	citeste un caracter dintr-un fisier
	fgets()	citeste un sir de caractere dintr-un fisier
	fscanf()	citeste date formatate dintr-un fisier
Functii diverse pentru fisiere	fseek()	pozitioneaza cursorul la un anumit octet in fisier
	feof()	returneaza true daca se ajunge la sfarsit de fisier
	ferror()	returneaza true daca a aparut o eroare
	rewind()	readuce indicatorul de pozitie la inceputul fisierului
	remove()	sterge un fisier
	fflush()	goleste un stream asociat unui fisier

6.4 Functii la nivel superior

Pointeri la fisiere

SINTAXA

Declarare pointer la fisier: FILE *pointer ;

unde: FILE este un cuvant rezervat in C/C++,
pointer este pointerul la fisier

FILE **=structura definita in** <stdio.h>

```
typedef struct{
    short level ; //Nivel plin/gol al bufferului
    unsigned flags;        //Indicatoare de stare fisier
    char fd;               //Descriptor fisier
    unsigned char hold
    short bsize; //Dimensiune buffer
    unsigned char *buffer          //Buffer transfer date
    unsigned char *curp            //Pointerul activ curent
    unsigned istemp       //Indicator de fisier temporar
    short token;           //testare validitate
    } FILE;
```

6.4 Functii la nivel superior

Deschidere fisier: fopen()

SINTAXA

```
FILE *pointer
pointer =fopen(const char *numefisier, const char *mod) ;
```

- **numefisier** este numele fisierului ce va fi deschis in modul “mod”,si poate include optional si o cale (director)
- **mod** este un sir de caractere care indica modul in care va fi deschis fisierul respectiv

EXEMPLU

Ex: Declarare variabila :

```
FILE  *fp;
fp = fopen( “fisier.dat", "r" );
```

unde: fisier.dat este fisierul ce va fi deschis pentru citire
fp este pointer la fisierul “fisier.dat”

6.4 Functii la nivel superior

Deschidere fisier: fopen()

FILE *fopen(const char *numefisier, const char *mod) ;

mod	Semnificatie
r	deschide un fisier text pentru citire
w	deschide/creeaza un fisier text pentru scriere
a	adauga intr-un fisier text
rb	deschide un fisier binar pentru citire
wb	creeaza un fisier binar pentru scriere
ab	adauga intr-un fisier binar
r+	deschide un fisier text pentru citire/scriere
w+	creeaza un fisier text pentru citire/scriere
a+	adauga in/creeaza un fisier text pentru citire/scriere
r+b	deschide un fisier binar pentru citire/scriere
w+b	creeaza un fisier binar pentru citire/scriere
a+b	adauga in/creeaza un fisier binar pentru citire/scriere

- r+b este echivalent cu rb+
- fopen() returneaza un pointer null in caz de eroare la deschiderea fisierului

6.4 Functii la nivel superior

Deschidere fisier: fopen()

EXEMPLE

Ex.1. : deschiderea fisierului test.txt pentru scriere

```
FILE *fp;
fp=fopen(”test.txt”,”w”);
```

Ex.2. : deschiderea fisierului test.txt pentru scriere , cu test erori

```
FILE *fp;
if ((fp=fopen(”test.txt”,”w”)==NULL)
    { printf(“nu se poate deschide fisierul.\n”); exit(1); }
```

Se va detecta orice eroare de deschidere a fisierului (ex: protectie la scriere, disc plin, etc.)

Ex.3. : deschiderea fisierului pentru scriere , cu test pentru erori

```
FILE *fis;
char s[20];
printf(“Introduceti numele fisierului: \n");
scanf("%s",s);
fis=fopen(s,"w");
if (fis==NULL)
    {printf("ERROR");exit(1);}
```

6.4 Functii la nivel superior

Deschidere fisier: fopen()

SINTAXA

Prototip functie: int fclose(FILE *fp) ;

- este inclusa in <stdio.h>
- fp este pointerul la fisierul deschis cu fopen()
- functia returneaza 0 in caz de succes, si EOF in caz de eroare

Efect: inchiderea stream-ului deschis catre fisierul respectiv

Fiecare fisier trebuie inchis separat

EXEMPLU

Ex. : deschiderea/inchiderea unui fisier

```
FILE *fp;
…
fp=fopen(”test.txt”,”w”);  //deschiderea fisierului
….                         //operatii de scriere in fisier
fclose(fp);                //inchiderea fisierului
```

6.4 Functii la nivel superior

Scrierea unui caracter intr-un fisier: fputc()

SINTAXA

Prototip functie int fputc(char ch, FILE *fp) ;

- este inclusa in <stdio.h>
- fp este pointerul returnat la deschiderea fisierului cu fopen() si specifica in ce fisier va fi scris caracterul ch
- ch este caracterul scris in fisier

EXEMPLU

Ex. : scrierea unui caracter in fisier

```
FILE *fp;
char ch="A";
…
fp=fopen("test.txt","w"); //deschiderea fisierului
fputc(ch, fp);            //operatie de scriere in fisier
fclose(fp);               //inchiderea fisierului
```

6.4 Functii la nivel superior

Scrierea unui caracter intr-un fisier: putc()

SINTAXA

Prototip functie int putc(char ch, FILE *fp) ;

- este inclusa in <stdio.h>
- fp este pointerul returnat la deschiderea fisierului cu fopen() si specifica in ce fisier va fi scris caracterul ch
- ch este caracterul scris in fisier

EXEMPLU

Ex. : scrierea unui caracter in fisier

```
FILE *fp;
char ch='A';
…
fp=fopen("test.txt","w"); //deschiderea fisierului
putc(ch, fp);             //operatie de scriere in fisier
fclose(fp);               //inchiderea fisierului
```

6.4 Functii la nivel superior

Citirea unui caracter dintr-un fisier: fgetc()

SINTAXA

Prototip functie: int fgetc(FILE *fp) ;

- este inclusa in <stdio.h>
- fp este pointerul returnat la deschiderea fisierului cu fopen()
- Functia returneaza caracterul citit convertit din unsigned char in tip int sau EOF in caz de eroare.

EXEMPLU

Ex.1 : citirea continutului unui fisier pina la sfirsit (EOF) si afisarea lui pe ecran

```
FILE *fp;
char ch;
…
fp=fopen(”test.txt”,”r”);        //deschiderea fisierului
do     {ch=fgetc(fp);
        while(ch!=EOF);    putchar(ch);            //printf("%c", ch);}
fclose(fp);                        //inchiderea fisierului
```

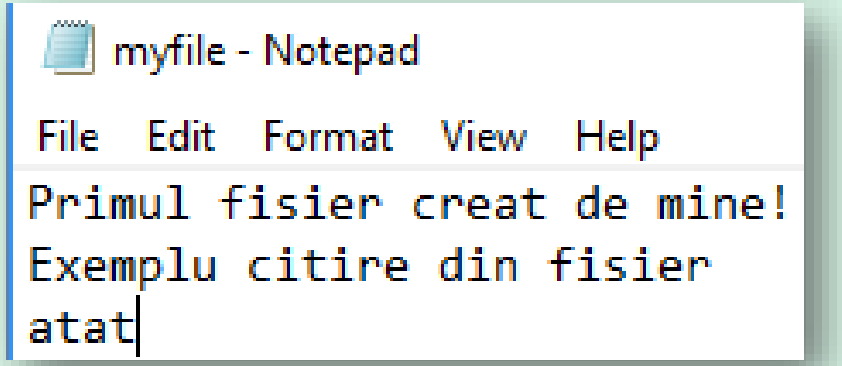
6.4 Functii la nivel superior

Citirea unui caracter dintr-un fisier: fgetc()

EXEMPLU

Ex.2: Programul numara liniiile de text din fisierul myfile.txt care **trebuie creat in directorul proiectului**

```
#include<stdio.h>
int main()
{ FILE *fp;
char ch;
int i=0;
fp = fopen("myfile.txt", "r");
while((ch=fgetc(fp))!=EOF)
      { if(ch == '\n')  i++; }
printf("In fisier sunt %d linii",i);
fclose(fp); return 0;}
```




```
In fisier sunt 3 linii
```

6.4 Functii la nivel superior

Citirea unui caracter utilizand streamuri standard: getc()

SINTAXA

Prototip functie: int getc(FILE *fp) ;

- este inclusa in <stdio.h>
- fp este pointerul returnat la deschiderea fisierului cu fopen()

Functia **returneaza caracterul citit sau EOF** in caz de eroare.

EXEMPLU

Ex. : citirea cate unui caracter de la intrarea standard si afisarea la iesirea standard

```
#include<stdio.h>
int main()
{ char c;
printf("Introduceti un caracter: ");
c = getc(stdin);
printf("Caracter  introdus: ");
putc(c, stdout);
return 0;}
```

```
Introduceti un caracter: c
Caracter  introdus: c
```

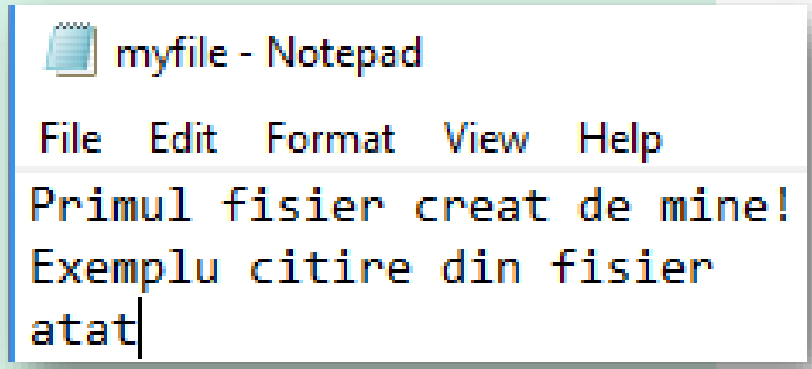
6.4 Functii la nivel superior

Citirea unui caracter dintr-un fisier: getc()

EXEMPLU

Ex. : Programul numara liniiile de text din fisierul myfile.txt care **trebuie creat in directorul proiectului**

```
#include<stdio.h>
int main()
{ FILE *fp;
char ch;
int i=0;
fp = fopen("myfile.txt", "r");
while((ch=getc(fp))!=EOF)
      { if(ch == '\n')  i++; }
printf("In fisier sunt %d linii",i);
fclose(fp); return 0;}
```




```
In fisier sunt 3 linii
```

6.4 Functii la nivel superior

Determinarea sfarsitului de fisier : feof()

SINTAXA

Prototip functie: int feof(FILE *fp) ;

- ❑ este definita in <stdio.h>, intr-un fisier binar primul caracter poate fi chiar EOF
- ❑ fp este pointerul returnat la deschiderea fisierului cu fopen()

Functia **returneaza 0 daca s-a ajuns EOF sau o val. >0** in caz contrar

EXEMPLU

Ex.1. : citirea continutului unui fisier pana la sfirsitul acestuia

```
while(!feof(fp)) ch=fgetc(fp);
```

Ex.2. : citirea din fisier cu mesaj de eroare

```
if( !feof ( fp))  printf("Sfarsit de fisier!\n");
```

Echivalent cu:

```
if( feof ( fp)==0)  printf("Sfarsit de fisier!\n");
```

6.4 Functii la nivel superior

Golirea unui stream: fflush()

SINTAXA

Prototip functie: int fflush(FILE *fp); int fflush (stdin);

- fp pointer la un fisier, functia se afla in <stdio.h>
- scrie continutul datelor din stream(buffer) in fisierul asociat lui fp
- daca fp este null atunci vor fi golite toate fisierele deschise pentru iesiri; returneaza 0 in caz de succes, altfel returneaza EOF

EXEMPLU

Ex. Compararea a 2 caractere preluate de la tastatura

```
#include <stdio.h>
void main()
{   char a,b;
    printf("Care caracter este mai mare?\n");
    printf("Introduceti un singur caracter:"); a=getchar(); fflush(stdin);
    printf("Introduceti un alt caracter:"); b=getchar(); fflush(stdin);
    if(a > b) {printf("'%c>%c\n",a,b);}
    else if (b > a) { printf("%c>%c\n",b,a);
    }   else  { printf("ati introdus acelasi caracter");  } }
```

```
Care caracter este mai mare?
Introduceti un singur caracter:d
Introduceti un alt caracter:a
d>a
```

6.4 Functii la nivel superior

Scrierea sirurilor de caractere in fisiere: fputs()

SINTAXA

Prototip functie: int fputs(const char *sir, FILE *fp) ;

- este inclusa in <stdio.h>
- functie similara cu puts() , scrie sirul de caractere ***sir** in fisierul specificat de ***fp**

Functia **returneaza o valoare pozitiva, sau EOF** ptr. eroare .

EXEMPLU

Ex.1 : scrie un sir de caractere intr-un fisier

```
char sir[80]="orice text" ;
FILE *fp;
fp=fopen("test.txt","w");
…
fputs(sir,fp);
```

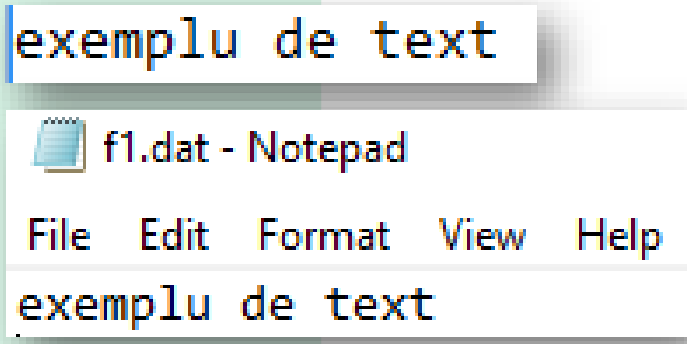
6.4 Functii la nivel superior

Scrierea sirurilor de caractere in fisiere: fputs()

EXEMPLU

Ex. 2 : citeste un sir de la tastatura si-l scrie intr-un fisier f1.dat

```
#include<stdio.h>
#include<string.h>
int main()
{ FILE *fptr;
char str[80];
fptr = fopen("f1.dat", "w");
if(fptr == NULL) printf("Nu pot deschide fisierul");
else {
      while(strlen(gets(str))>0) //cat timp sirul citit de la tastatura >0
              { fputs(str, fptr);  //scrie sirul citit in fisierul f1.dat
                fputs("\n", fptr); } //adauga un enter la fiecare sir
fclose(fptr); }
return 0;}
```



6.4 Functii la nivel superior

Citirea sirurilor de caractere : fgets()

SINTAXA

Prototip functie char *fgets(char *sir, int n, FILE *fp)

- este definita in <stdio.h>
- fgets() **citeste un sir de lungime n caractere din fisierul specificat** de fp si-l memoreaza la adresa indicata de pointerul sir.

Functia **returneaza pointer la un sir sau un pointer NULL** in caz de eroare

EXEMPLU

Ex. : citeste cate un sir dintr-un fisier in.txt si-l scrie in fisierul out.txt

```
char sir[80] ;
FILE *fp1, *fp2;
fp1=fopen("in.txt","r");
fp2=fopen("out.txt","w"); …
 while(fgets(sir,sizeof(sir),fp1) //citire din in.txt
        fputs(sir,fp2);                    //scriere in out.txt
```

6.4 Functii la nivel superior

Scrierea datelor cu format in fisiere: fprintf()

SINTAXA

Prototip functie: int fprintf(FILE *fp, const char *sir_control,…);

- este definita in <stdio.h>
- fp este un pointer de fisier, returnat de o apelare a functiei fopen();
- operatia de scriere se efectueaza asupra acestui fisier.

Efect: scrie in fisierul specificat prin pointerul fp date formatate

EXEMPLU

Ex. : scrierea intr-un fisier a 2 variabile, tip sir si respectiv tip intreg

```
FILE *fp;
char s[80];
int t;
….
fp=fopen(“date.txt”, “w”)
fprintf(fp,”Sirul este:%s, n=%d”, s,t ); //scrie in fisier
```

6.4 Functii la nivel superior

Citirea datelor cu format din fisiere: fscanf()

SINTAXA

Prototip functie: int fscanf(FILE *fp, const char *sir_control,…);

- este definita in <stdio.h>
- fp este un pointer de fisier, returnat de o apelare a functiei fopen(), iar operatia de citire se efectueaza asupra acestui fisier.

Efect: citeste din fisierul specificat prin pointerul fp date formatate

EXEMPLU

Ex. : operatii citire/scriere in acelasi fisier

```
FILE *fp;
char s[80];int t;
….
fscanf(fp, “%s,%d”,s,&t); //citeste din fisier un sir si un intreg
…
fprintf(fp,”Sirul este:%s, n=%d”, s,t ); //scrie in fisier cele 2 variabile
```

6.4 Functii la nivel superior

Repozitionarea indicatorului de pozitie: rewind()

SINTAXA

Prototip functie: void rewind(FILE *fp);

- fp pointer la un fisier,
- functia este definita in <stdio.h>

Efect: readuce indicatorul de pozitie la inceputul fisierului

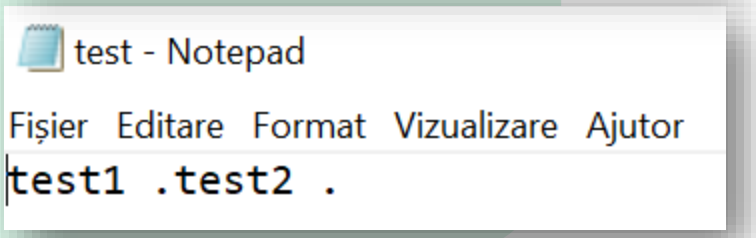
Ex. : citeste de la tastatura un sir si il scrie/adauga in fisier, apoi dupa rewind , citeste din fisier si afiseaza pe monitor continutul fisierului

EXEMPLU

```
#include <stdio.h>
#include<string.h>
int main()
{FILE *fp; char s[20];
fp=fopen("test.txt","a+");
printf("Introduceti un sir terminat cu Enter:");
gets(s); fputs(s,fp);fputs(" .",fp);
rewind(fp);
while(!feof(fp)){ fgets(s,20,fp); puts(s);}
return 0;}
```

```
Introduceti un sir terminat cu Enter:test1
test1 .

Introduceti un sir terminat cu Enter:test2
test1 .test2 .
```



6.4 Functii la nivel superior

Determinarea unei erori: ferror()

SINTAXA

Prototip functie: int ferror(FILE *fp);

unde fp pointer la un fisier, functia se afla in <stdio.h>

Efect: returneaza o valoare pozitiva daca a aparut o eroare in timpul ultimei operatii asupra fisierului, **sau 0** in caz de reusita.

EXEMPLU

Ex. :testarea cu mesaj de eroare la citirea dintr-un fisier

```
…
FILE *fp;
char ch;
fp=fopen(“test.a”,”r”);
…
ch=fgetc(fp)
if(ferror(fp)) {printf (“eroare la citirea din fisier”); exit(1);}
…
```

6.4 Functii la nivel superior

Stergerea fisierelor: remove()

SINTAXA

Prototip functie: int remove(const char *numefisier);

- sterge fisierul specificat. **Returneaza 0 succes** operatie de stergere , **altfel ≠ 0**

EXEMPLU

Ex. : stergerea unui fisier specificat in linia de comanda

```
//programul  executabil sterge.exe
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
int main(int argc,char *argv[])
{ char c,s[50];
   if (argc!=2){ printf("corect sterge fisier.extensie\n");exit(1);}
   printf("Sterg %s ? (Y/N):",argv[1]);   getc(c);
   if(toupper(c)=='Y')
   if(!remove(argv[1])) { printf("fisierul nu se poate sterge\n");exit(1);}
   else printf("Chiar l-am sters !"); return 0; }
```

```
>sterge text.txt
>Sterg text.txt ?(Y/N): y
>Chiar l-am sters!
```

Rezultate afisate:

```
>sterge
>corect sterge fisier.extensie
```

Rezultate afisate:

```
>sterge text1.txt
>fisierul nu se poate sterge
```

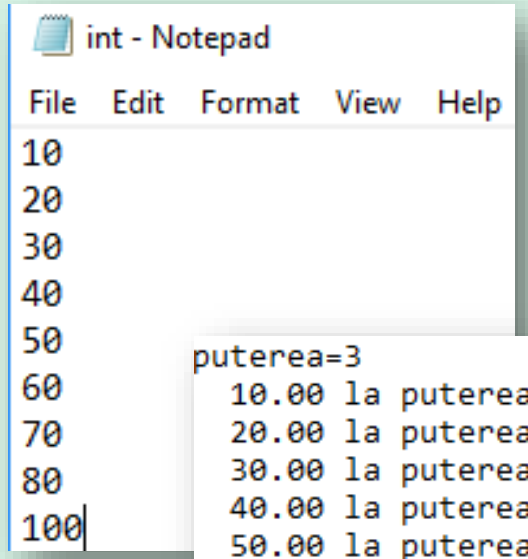
6.4 Functii la nivel superior

Exemple operatii cu fisiere

EXEMPLU

Ex.1. Se editeaza un fisier numit int.txt care contine un sir de numere reale x. Se citeste valoarea reala y=puterea de la tastatura si se tiparesc valorile x la puterea y in out.txt.

```
#include <stdio.h>
#include <math.h>
int main (void)
{FILE *fis1;  FILE *fis2;
double x,y,p;
printf("puterea=");scanf("%lf", &y);
fis1=fopen("int.txt","r");
fis2=fopen("out.txt","w");
while (fscanf(fis1,"%lf",&x)!=EOF)
   {  p=pow(x,y);
printf("%7.2lf la puterea %7.2lf-a =%7.2lf \n", x, y,p);
fprintf(fis2,"%7.2lf la puterea %7.2lf-a =%7.2lf \n", x,y,p);}
fclose (fis1);
fclose (fis2);return 0;}
```




```
puterea=3
  10.00 la puterea     3.00-a =1000.00
  20.00 la puterea     3.00-a =8000.00
  30.00 la puterea     3.00-a =27000.00
  40.00 la puterea     3.00-a =64000.00
  50.00 la puterea     3.00-a =125000.00
  60.00 la puterea     3.00-a =216000.00
  70.00 la puterea     3.00-a =343000.00
  80.00 la puterea     3.00-a =512000.00
 100.00 la puterea     3.00-a =1000000.00
```

6.4 Functii la nivel superior

Exemple operatii cu fisiere

EXEMPLU

Ex.2: Se citesc pe rand cate o operatie din fisierul a.txt, respectiv op1 op op2 si se tipareste rezultatul operatiei in fisierul ab.txt

```
#include <stdio.h>
int main (void)
{FILE *fis1,*fis2;
int op1,op2,rez; char op;
fis1=fopen("a.txt","r");//fisier intrare:se citesc datele
fis2=fopen("ab.txt","wa");//fisier iesire:se scriu datele
while (fscanf(fis1,"%d%c%d\n",&op1,&op,&op2)!=EOF)
{switch (op) {
case '+': rez=op1+op2; break;
case '-': rez=op1-op2; break;
case '*': rez=op1*op2; break;
case '/': if (op2==0) { rez=0;fprintf(fis2,"divizor nul, rez=%d ", rez); }
          else {rez=op1/op2;  break;}
default: rez=0; fprintf(fis2,"operator eronat:");}
     fprintf(fis2,"%d%c%d=%d\n",op1,op,op2,rez);}
     printf("Fisierul rezultat a fost creat ");return 0;}
```

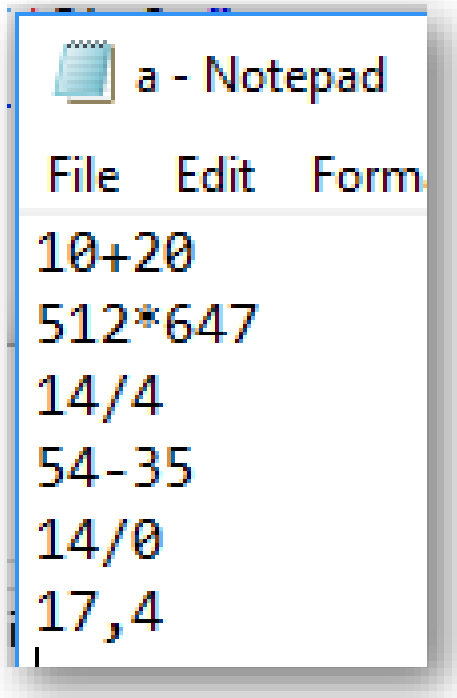
Cum adaug operatia x^2+y^2?

```
#include<math.h>
case '^': rez=pow(op1,2)+pow(op2,2); break;
```

6.4 Functii la nivel superior

Exemple operatii cu fisiere

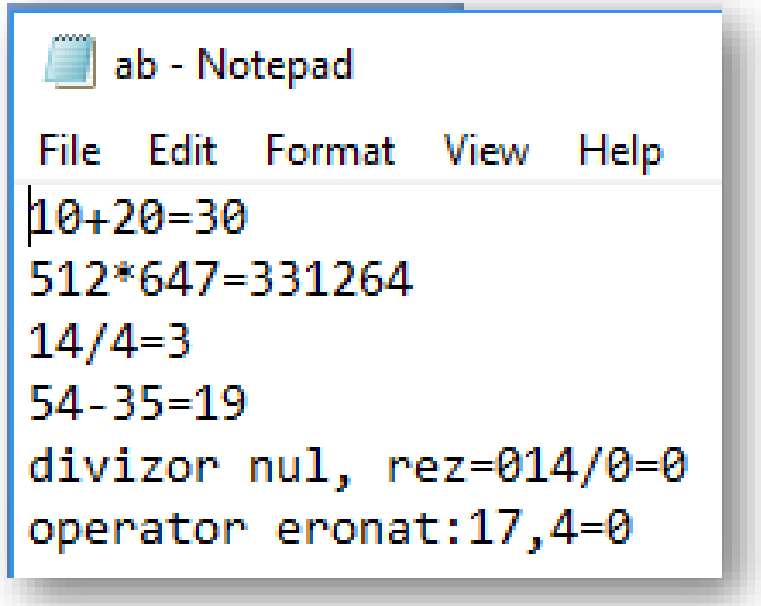
a.txt



fisier intrare:din care se citesc datele

EXEMPLU

ab.txt



fisier iesire:in care se scriu datele

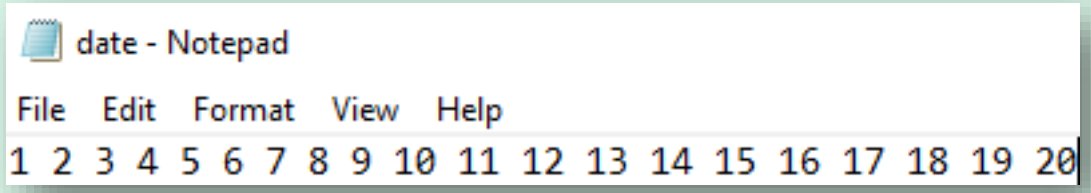
6.4 Functii la nivel superior

Exemple operatii cu fisiere

EXEMPLU

Ex.3: Se citeste cate un numar dintr-un sir de **20 nr intregi** dintr-un fisier si se calculeaza si afiseaza suma lor

```
#include<stdio.h>
int main()
{int num[20];
int i = 0, tot = 0;
FILE *fptr;
fptr = fopen("date.txt", "r");
if(fptr == NULL){printf("File not exist");}
for(i = 0; i < 20; i++)
    {fscanf(fptr, "%d ", &num[i]);
    printf("%d ", num[i]); tot+= num[i];}
printf("\ntotal = %d", tot);
fclose(fptr); return 0;}
```




```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
total = 210
```

Cum adaug produsul numerelor?

```
int prod=1;
```

…

in for() se adauga `prod*=num[i];`

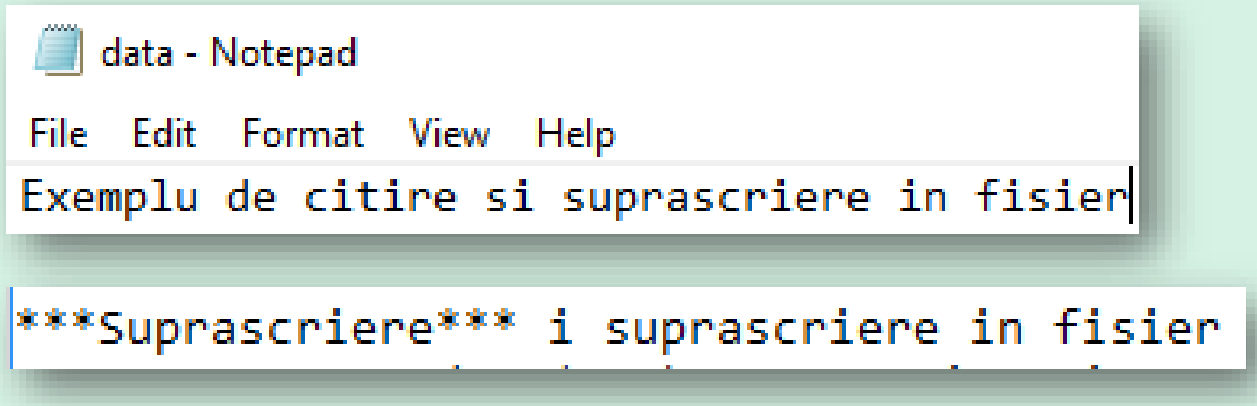
6.4 Functii la nivel superior

Exemple operatii cu fisiere

EXEMPLU

Ex.4: Se deschide un fisier data.txt pentru citire si scriere cu “r+”. In fisier initial este scris textul “Exemplu de citire si suprascriere in fisier ” apoi se scrie textul "***Suprascriere***” peste textul existent

```
#include<stdio.h>
int main()
{char ch;
FILE *fptr;
fptr = fopen("data.txt", "r+");
fprintf(fptr, "***Suprascriere*** ");
fclose(fptr);
fptr = fopen("data.txt", "r");
while((ch = fgetc(fptr))!= EOF)
     {       printf("%c",ch);     }
fclose(fptr);return 0;}
```



TEST

1 . Care este efectul instructiunilor ?

```
FILE *fis1;
int i;
fis1=fopen("in.txt","w");
for (i=1;i<5;i++)
fprintf(fis1,"%d,",2*i+1); fclose (fis1);
```

se citesc din fişierul in.txt numerele intregi 1,2,3,4
se scriu in fisierul in.txt numerele intregi 3,5,7,9
se afiseaza pe ecran numerele intregi 3,5,7,9
se scriu in fisierul in.txt numerele intregi 1,2,3,4

Raspuns correct

b)

TEST

2. Care este efectul instructiunilor ?

```
FILE *p; int i;
 p=fopen("rez.txt", "r");
 while (fscanf(p,"%d",&i)!=EOF) printf("%d ",i+3);
 fclose (p);
```

daca fisierul rez.txt contine valorile intregi 1,2 si 3.

a) se citesc din fisierul rez.txt numerele 1,2,3 si se afiseaza pe ecran valorile 1,2,3
b) se citesc din fisierul rez.txt numerele 1,2,3 si apoi se scriu in acelasi fisier numerele 4,5, 6
c) se scriu in fisierul rez.txt valorile 1,2 si 3 si apoi se scriu in acelasi fisier numerele 4,5, 6
d) se citesc din fisierul rez.txt numerele 1, 2, 3 si apoi se afiseaza pe ecran valorile 4, 5 si 6

Raspuns correct

d)

6.4 Functii la nivel superior

Fisiere binare. Scrierea /citirea blocurilor de date: fread(), fwrite()

SINTAXA

Prototip functie:

size_t fread(void *buffer, size_t nrocteti, size_t numar, FILE *fp);

- buffer= pointer catre o regiune de memorie care va primi date de la fisier
- numar= **nr. de elemente citite**, avand un **nr. de octeti** =nrocteti
- size_t este definit in <stdio.h> si este aprox. echiv cu **intreg fara semn**

Efect: functia returneaza nr. de elemente citite din fisierul binar;

Aceasta valoare poate fi mai mica decat numar daca se ajunge la sf. fisierului sau daca apare o eroare; Este definita in <stdio.h>

Prototip functie:

size_t fwrite(const void *buffer, size_t nrocteti, size_t numar, FILE *fp);

- buffer=pointer catre informatiile care vor fi scrise in acel fisier
- numar= **nr. de elemente scrise**, avand un **nr. de octeti** =nrocteti

Efect: functia returneaza nr. de elemente scrise in fisier;

Aceasta valoare va fi egala cu numar daca nu apare o eroare; Este definita in <stdio.h>

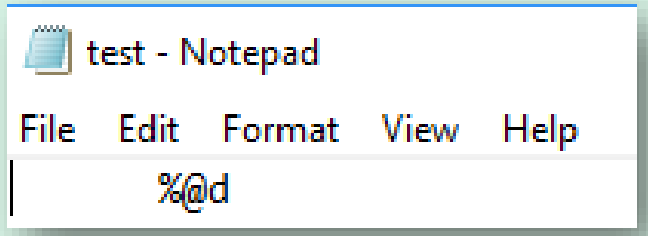
6.4 Functii la nivel superior

Fisiere binare. Scrierea /citirea blocurilor de date: fread(), fwrite()

EXEMPLU

Ex. : scrierea/citirea unor variabile tip int si double in acelasi fisier binar

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{FILE *fp;
double d=10.5;
int i=100;
if((fp=fopen ("test","wb+"))==NULL) {          printf("nu se poate deschide fisierul\n"); exit(1); }
fwrite(&d, sizeof(double),1,fp);
fwrite(&i, sizeof(int),1,fp);rewind(fp);
fread(&d, sizeof(double),1,fp);
fread(&i, sizeof(int),1,fp);
printf("%lf %d ",d,i);fclose(fp); //afisam pe ecran d si i
return 0;}
```




```
10.500000 100
```

6.4 Functii la nivel superior

Acces aleator la fisier: fseek()

SINTAXA

Prototip functie: int fseek(FILE *fp, long numocteti, int origine);

- definita in <stdio.h>
- fp este un pointer pentru fisier, returnat de o apelare a functiei fopen();
- numocteti corespunde nr. de octeti de la origine care va deveni noua pozitie curenta
- origine este una din urmatoarele definitii macro din <stdio.h>:
  - Inceput fisier SEEK_SET
  - Pozitie curenta SEEK_CUR
  - Sfarsit fisier SEEK_END

Efect: **functia returneaza 0 pentru operatie incheiata cu succes**, si valoare !0 pentru eroare. S**e utilizeaza pentru citire/scriere aleatorie**

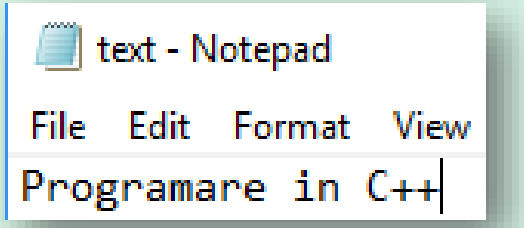
6.4 Functii la nivel superior

Acces aleator la fisier: fseek()

EXEMPLU

Ex. : cautarea unui octet intr-un fisier de intrare . Linia de comanda :
>fseek text.txt 0

```
#include <stdio.h>
#include <stdlib.h>
int main (int argc,char *argv[]) //fseek.exe
{FILE *fp;
if(argc!=3){ printf("fseek numefisier octet \n"); exit(1);}
if((fp=fopen(argv[1],"r"))==NULL){ printf("nu pot deschide fisierul"); exit(1);}
if(fseek(fp, atoi(argv[2]),SEEK_SET)) {printf("Eroare citire din fisier");exit(1);}
printf("La octetul %d este %c.\n", atoi(argv[2]), getc(fp));
return 0;}
```




```
D:\laura\codeblocks\p0\bin\Debug>fseek text.txt 0
La octetul 0 este P.

D:\laura\codeblocks\p0\bin\Debug>fseek text.txt 1
La octetul 1 este r.

D:\laura\codeblocks\p0\bin\Debug>fseek text.txt 5
La octetul 5 este a.
```

TEST kahoot

Pentru login, introduceti codul afisat pe ecran, in browser la adresa:

http://kahoot.it