Chapter 8:Files & File Handeling

Er. Krishna Khadka

Introduction

- In previous units, we used the function scanf() to read data from keyboard and printf() to display the data on screen.
- In such programs, data is temporarily stored in main memory during the program execution.
- When the program is re-complied, the entered data have been lost.
- Therefore, in order to permanently store data, we need data file.
- A data file is simply a file in hard disk which is used to store data permanently.

Functions in file handling

- language supports number of functions to perform various operations such as defining file, opening file, reading file, writing file, appending file and closing file.

fopen()	To create new data file
fclose()	To close data file
getc()	To read a character from data file
putc( )	To write a character to data file
getw()	To read an integer from data file

Functions in file handling

putw()	To write an integer to data file
fscanf()	To read formatted data from data file
fprintf()	To write formatted data to data file
fread()	To read record from data file
fwrite()	To write record to data file

Functions in file handling

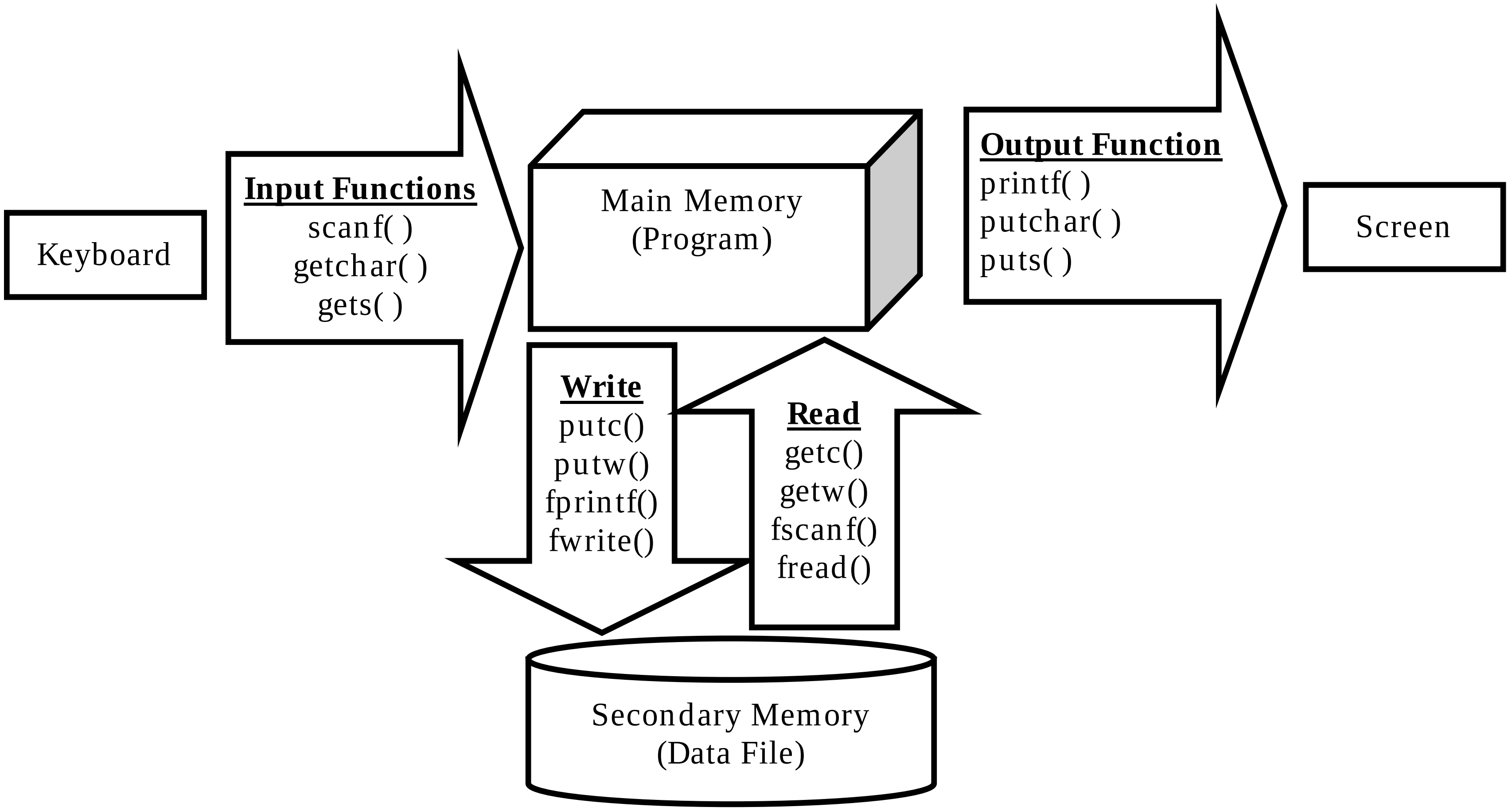


Figure: Input /output functions and read/write function

Defining and Opening Data File

- The following syntax is used to define data file

```
        FILE *fp;
        fp= fopen("filename","mode");
```

- The keyword FILE is used to declare file data type.
- The first statement defines file pointer variable with data type FILE.
- The second statement creates file with name filename and returns memory address that is assigned to the file pointer fp.

The mode defines the purpose of the file which can be one of the followings:

r	open an existing file for reading only.
w	open a new file for writing only. If a file with the specified file name currently exists, it will be destroyed and a new file will be created in its place.
a	open an existing file for appending (i.e. for adding new information at the end of the file). A new file will be created if the file with the specified file name does not exist.
r+	open an existing file for both reading and writing.
w+	open a new file for both reading and writing. If a file with the specified file name currently exists, it will be destroyed and a new file will be created in its place.
a+	open an existing file for both reading and appending. A new file will be created if the file with the specified file name does not exist.

Compiler automatically assigns special character at the end of the file called EOF (end of file).

Closing Data File

- A data file must be closed after doing various operations such as reading, writing etc.
- It ensures that all the information associated with the file is flushed out from the memory and all links to the file are broken.
- It also helps us to protect from accidental lose of data from the file.
- The following syntax is used to close the file.

```
fclose(filepointer);
```

Algorithm (Opening file for writing)

- **Step 1:** Define file pointer variable with type FILE either inside or outside main function.

  `FILE *file_pointer;`

- **Step 2:** Open file in write or append mode.

  `file_pointer =fopen(“ file_name”, “w”or “a”);`

- **Step 3:** Perform general I/O operations
- **Step 4:** Write data into file using following syntax:

  `fprintf(file_pointer, “format_specifiers”, variables);`

- **Step 5:** Close the file using following syntax:

  `fclose(file_pointer);`

Algorithm (Opening file for reading)

- **Step 1:** Define file pointer variable with type FILE either inside or outside void main function.

```
FILE *file_pointer;
```

- **Step 2:** Open file in read mode.

```
file_pointer =fopen(“ file_name”, “r”);
```

- **Step 3:** Read data from file using following syntax:

```
fscanf(file_pointer, “format_specifiers”, variables);
```

- **Step 4:** Display the data as required
- **Step 5:** Close the file using following syntax:

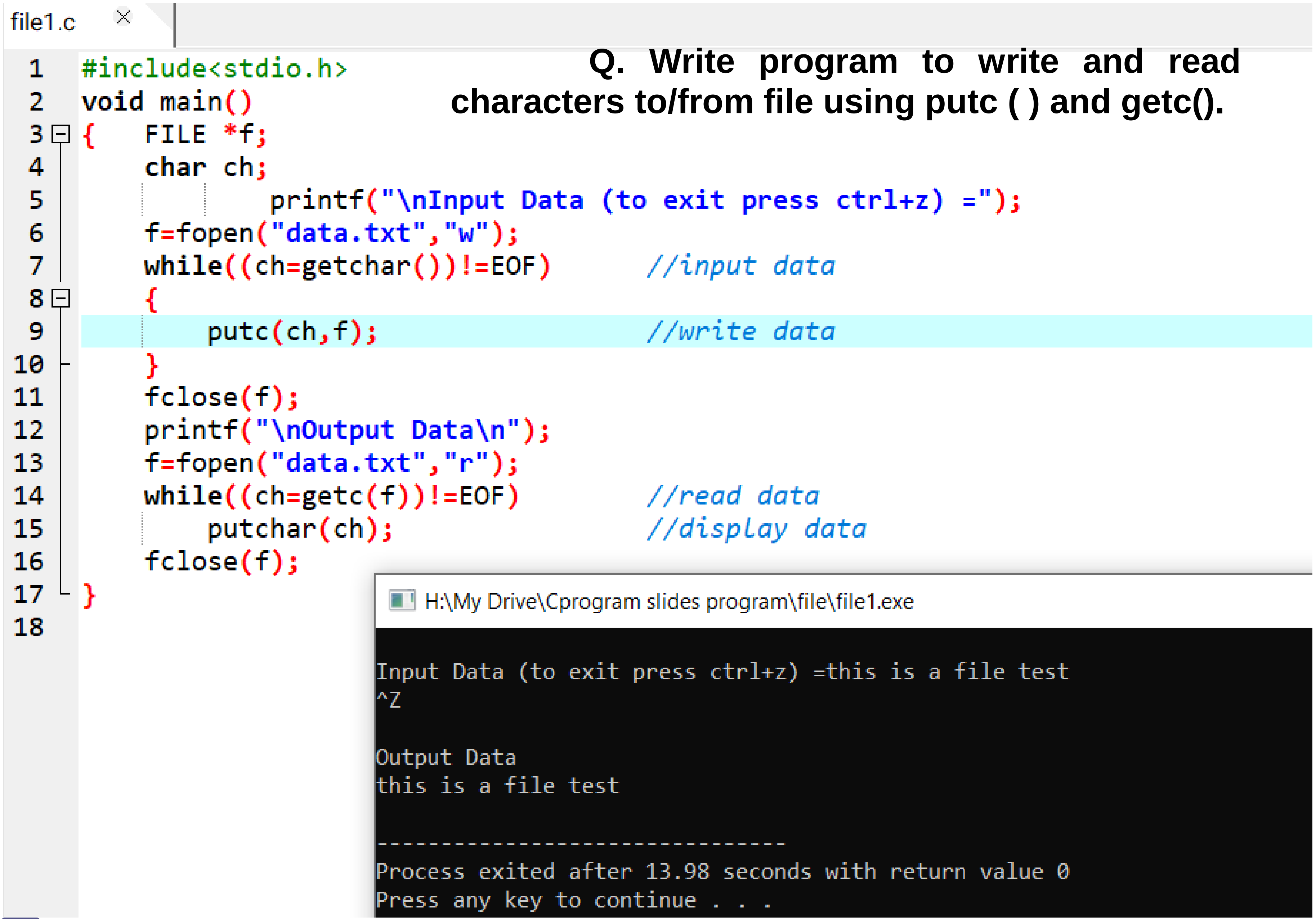
```
fclose(file_pointer);
```

getc() and putc( ) functions

- Function getc() helps us to read single character from file and putc( ) helps us to write single character to file.
- In the following example, ch is character variable and fp is file pointer.

```
ch= getc(fp);  //  to read data
putc(ch,fp);   //to write data, fp is file pointer and
               ch is character variable
```

Q. Write program to write and read characters to/from file using putc ( ) and getc().



file1.c

```c
#include<stdio.h>
void main()
{   FILE *f;
    char ch;
            printf("\nInput Data (to exit press ctrl+z) =");
    f=fopen("data.txt","w");
    while((ch=getchar())!=EOF)      //input data
    {
        putc(ch,f);                 //write data
    }
    fclose(f);
    printf("\nOutput Data\n");
    f=fopen("data.txt","r");
    while((ch=getc(f))!=EOF)        //read data
        putchar(ch);                //display data
    fclose(f);
}
```

H:\My Drive\Cprogram slides program\file\file1.exe

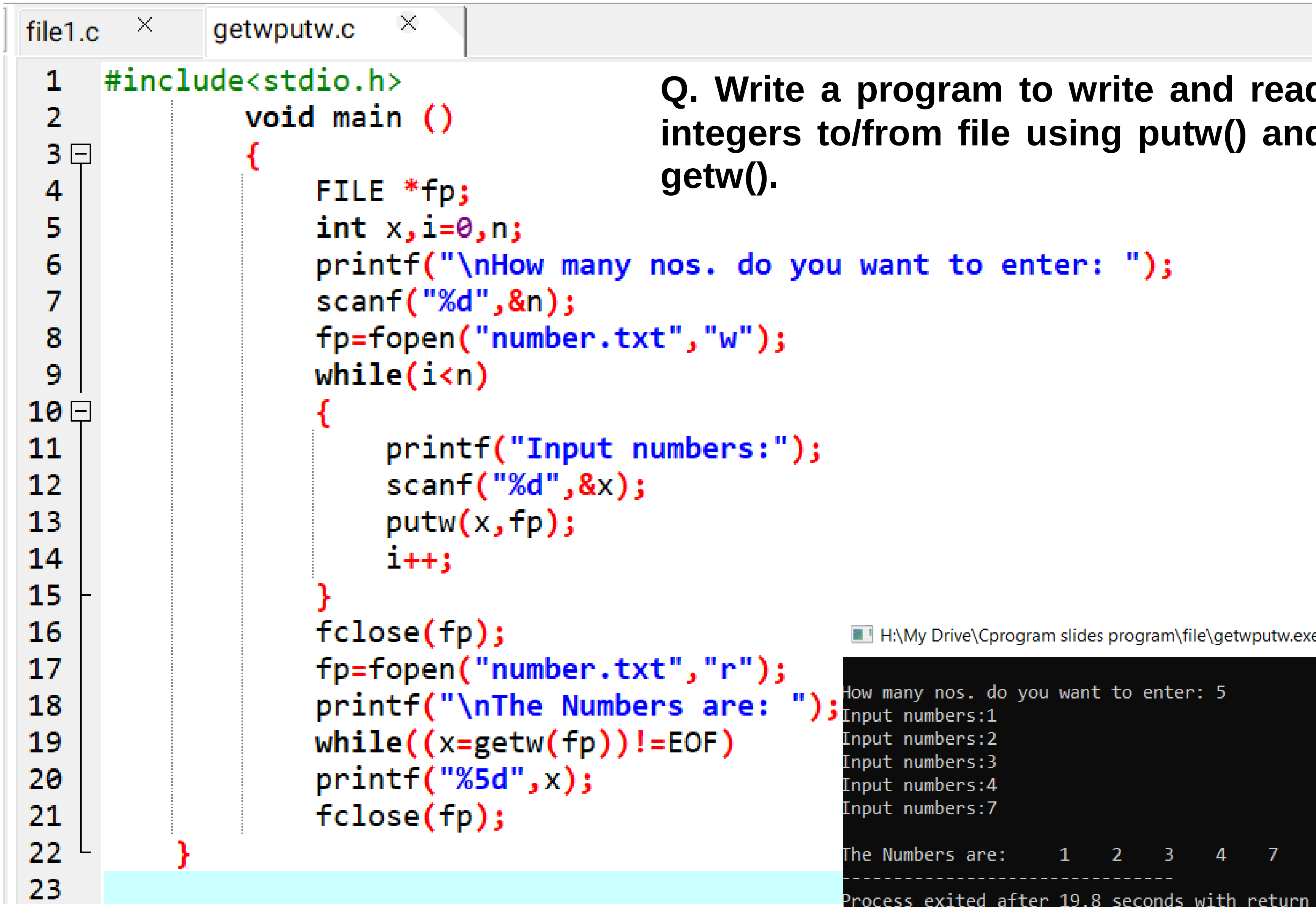
```
Input Data (to exit press ctrl+z) =this is a file test
^Z

Output Data
this is a file test


--------------------------------
Process exited after 13.98 seconds with return value 0
Press any key to continue . . .
```

getw() and putw() functions

- Function getw() helps us to read single integer from file and putw() helps us to write single integer to the data file.
- In the following example, x is integer variable and fp is file pointer.

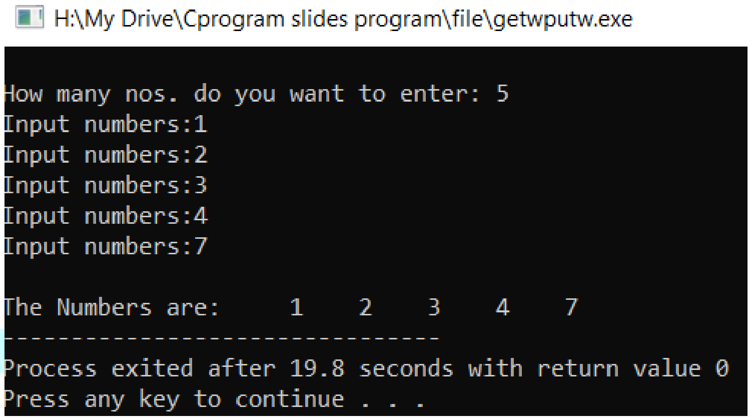
```
x = getw(fp);
putw(x,fp);
```

Q. Write a program to write and read integers to/from file using putw() and getw().



file1.c getwputw.c

```c
#include<stdio.h>
        void main ()
        {
            FILE *fp;
            int x,i=0,n;
            printf("\nHow many nos. do you want to enter: ");
            scanf("%d",&n);
            fp=fopen("number.txt","w");
            while(i<n)
            {
                printf("Input numbers:");
                scanf("%d",&x);
                putw(x,fp);
                i++;
            }
            fclose(fp);
            fp=fopen("number.txt","r");
            printf("\nThe Numbers are: ");
            while((x=getw(fp))!=EOF)
            printf("%5d",x);
            fclose(fp);
        }
```



H:\My Drive\Cprogram slides program\file\getwputw.exe

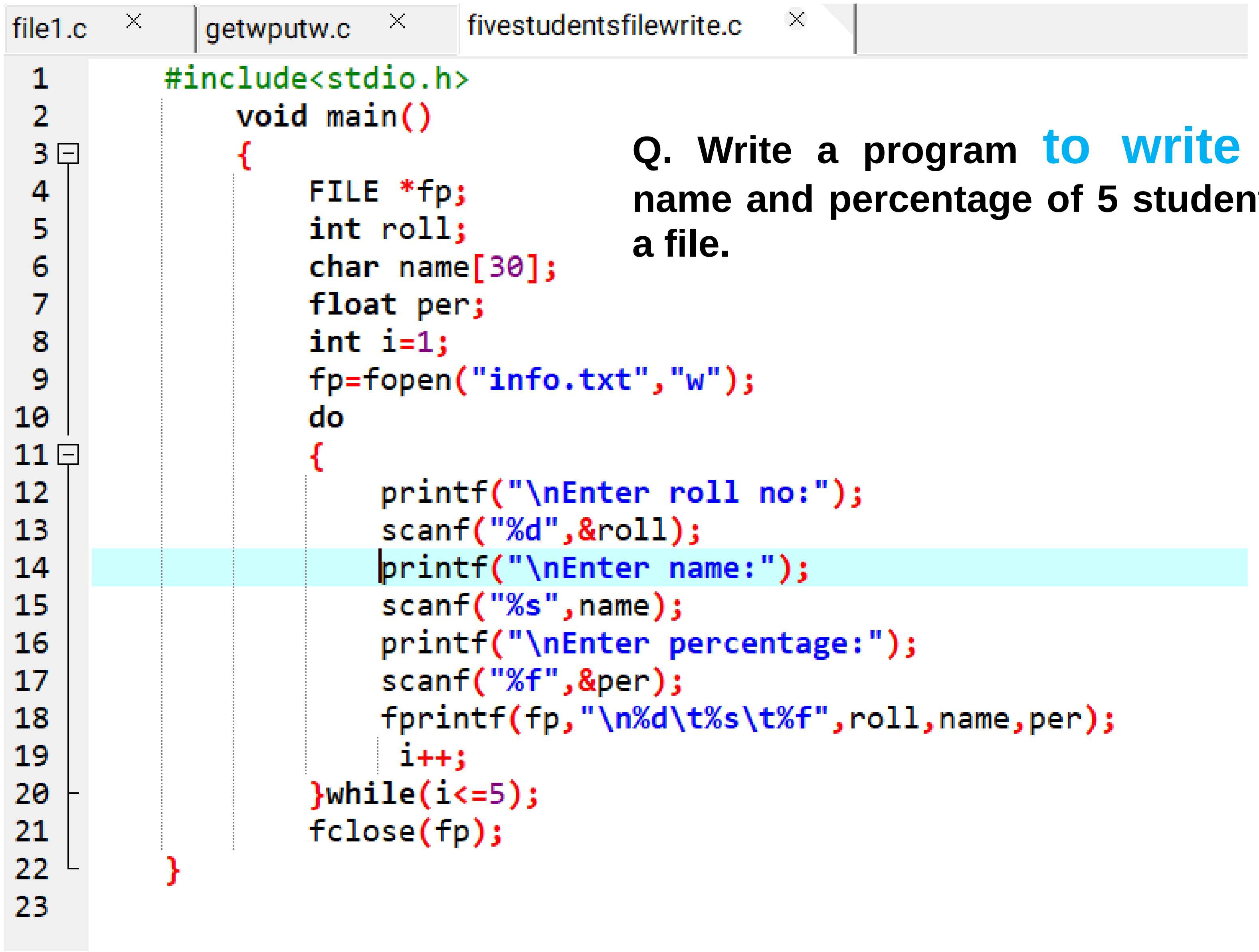
```
How many nos. do you want to enter: 5
Input numbers:1
Input numbers:2
Input numbers:3
Input numbers:4
Input numbers:7

The Numbers are:     1    2    3    4    7
--------------------------------
Process exited after 19.8 seconds with return value 0
Press any key to continue . . .
```

fscanf() and fprintf() funtions

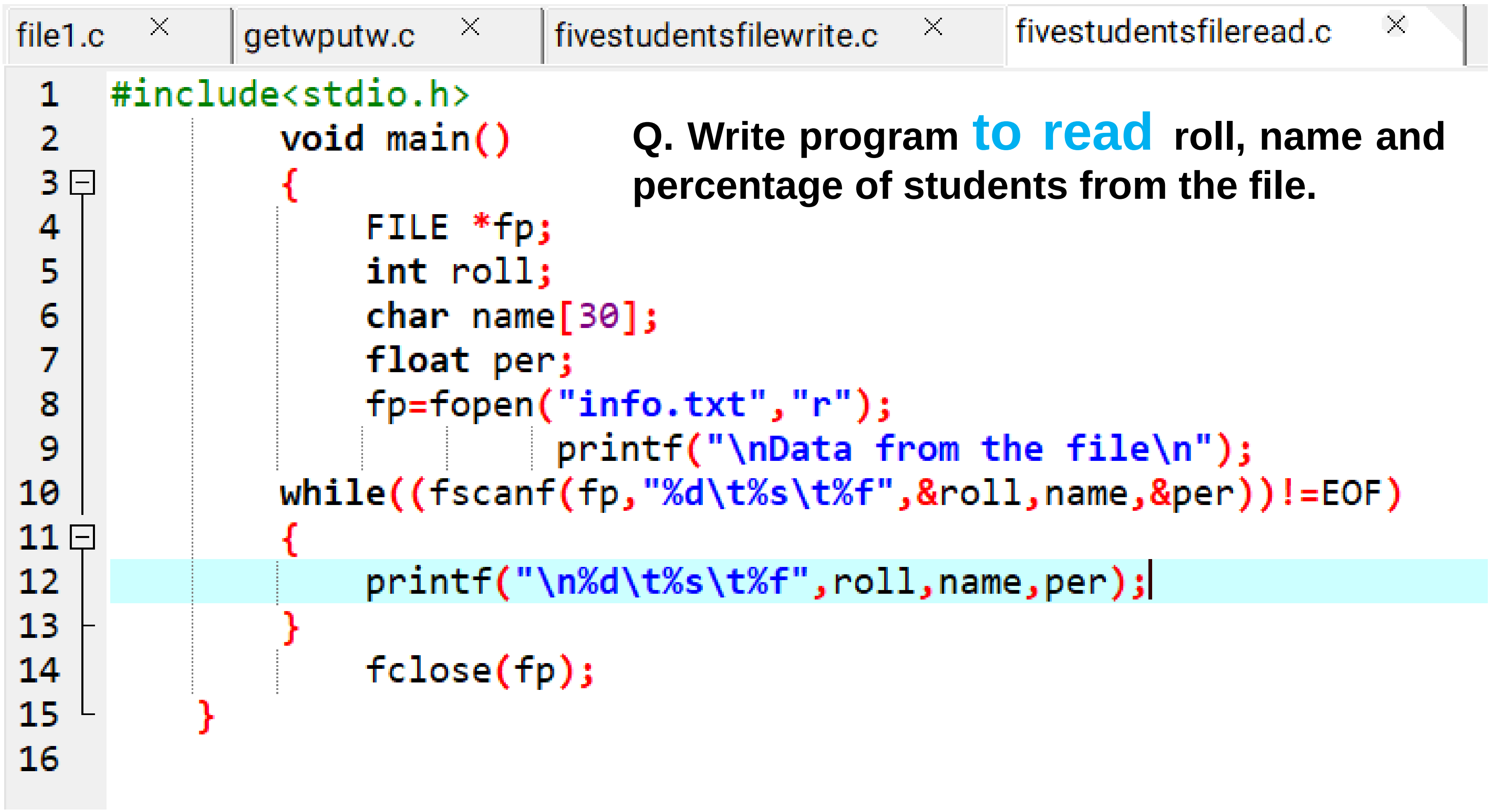
- Function fscanf() helps us to read formatted data from file and fprintf() helps us to write formatted data to file.
- In the following example, fp is file pointer, control string defines data type for the respective variables.

```
fscanf(fp,“control string”,variables);
fprintf(fp,“control string”,variables);
```

Q. Write a program to write roll, name and percentage of 5 students to a file.


file1.c getwputw.c fivestudentsfilewrite.c

```
#include<stdio.h>
    void main()
    {
        FILE *fp;
        int roll;
        char name[30];
        float per;
        int i=1;
        fp=fopen("info.txt","w");
        do
        {
            printf("\nEnter roll no:");
            scanf("%d",&roll);
            printf("\nEnter name:");
            scanf("%s",name);
            printf("\nEnter percentage:");
            scanf("%f",&per);
            fprintf(fp,"\n%d\t%s\t%f",roll,name,per);
             i++;
        }while(i<=5);
        fclose(fp);
}
```


file1.c × getwputw.c × fivestudentsfilewrite.c × fivestudentsfileread.c ×

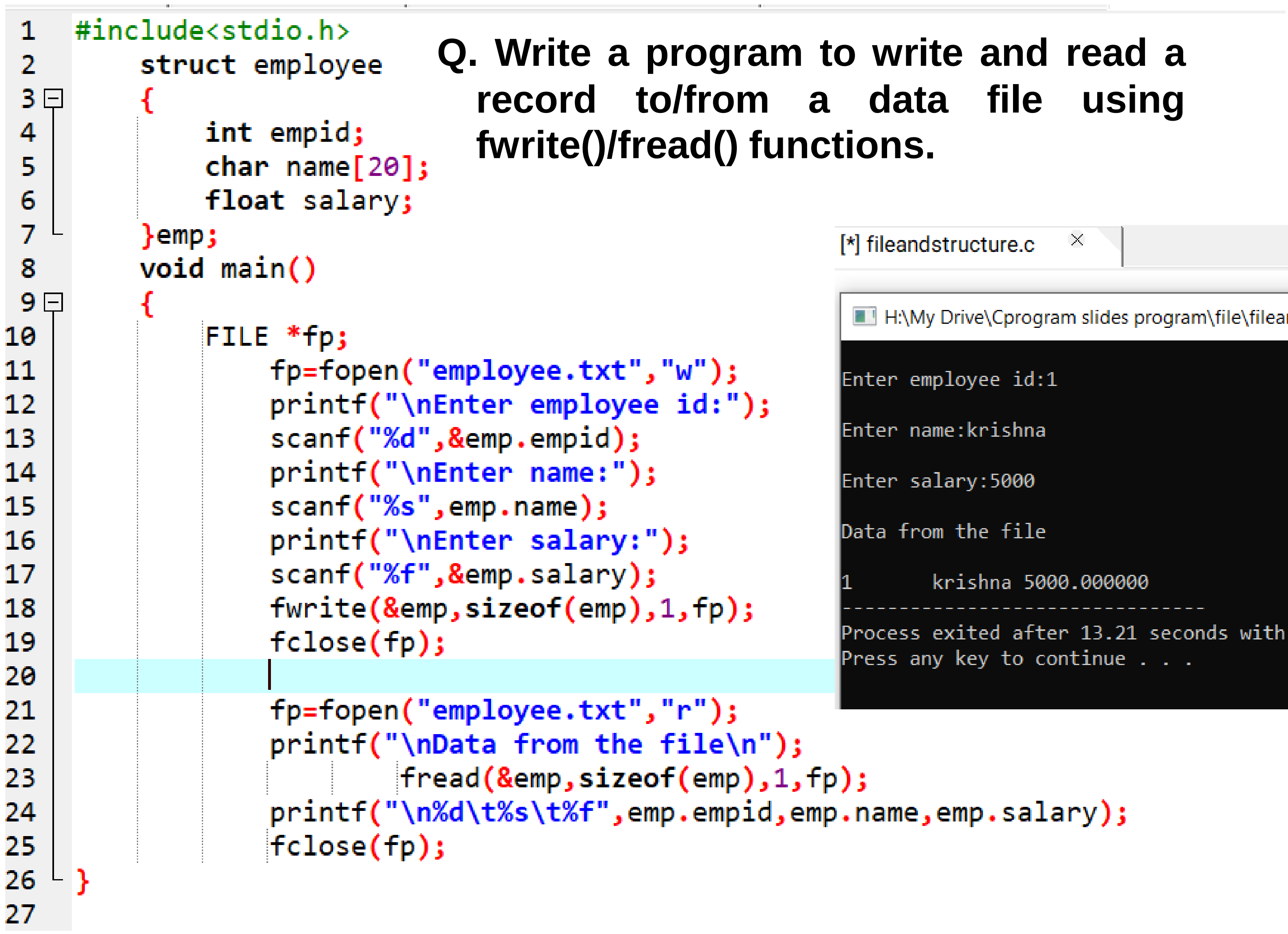
```c
#include<stdio.h>
	void main()
	{
		FILE *fp;
		int roll;
		char name[30];
		float per;
		fp=fopen("info.txt","r");
				printf("\nData from the file\n");
	while((fscanf(fp,"%d\t%s\t%f",&roll,name,&per))!=EOF)
	{
		printf("\n%d\t%s\t%f",roll,name,per);
	}
		fclose(fp);
}
```

Q. Write program to read roll, name and percentage of students from the file.

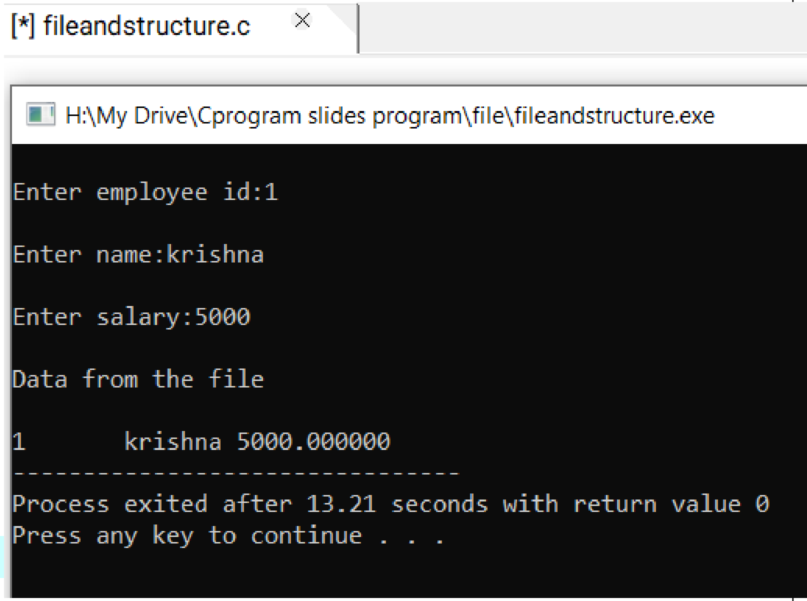
fread() and fwrite() functions

- Function fread() helps us to read structure type data i. e. record from data file and function fwrite() helps us to write structure type data to the data file.
- In the following example, variable is structure type variable of type tagname,
- sizeof() functions determines the memory size of structure,
- 1 specifies the number of record that can be accessed at once and fp is file pointer.
- `fread(&variable, sizeof(tagname),1,fp);`
- `fwrite(&variable, sizeof(tagname),1,fp);`

Q. Write a program to write and read a record to/from a data file using fwrite()/fread() functions.



```c
#include<stdio.h>
    struct employee
    {
        int empid;
        char name[20];
        float salary;
    }emp;
    void main()
    {
        FILE *fp;
            fp=fopen("employee.txt","w");
            printf("\nEnter employee id:");
            scanf("%d",&emp.empid);
            printf("\nEnter name:");
            scanf("%s",emp.name);
            printf("\nEnter salary:");
            scanf("%f",&emp.salary);
            fwrite(&emp,sizeof(emp),1,fp);
            fclose(fp);

            fp=fopen("employee.txt","r");
            printf("\nData from the file\n");
                    fread(&emp,sizeof(emp),1,fp);
            printf("\n%d\t%s\t%f",emp.empid,emp.name,emp.salary);
            fclose(fp);
}
```

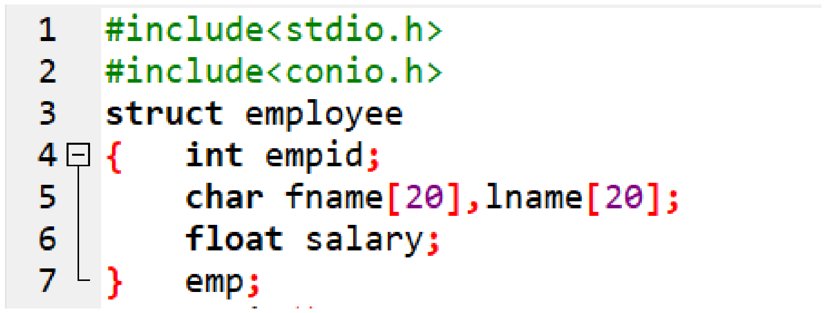


[*] fileandstructure.c

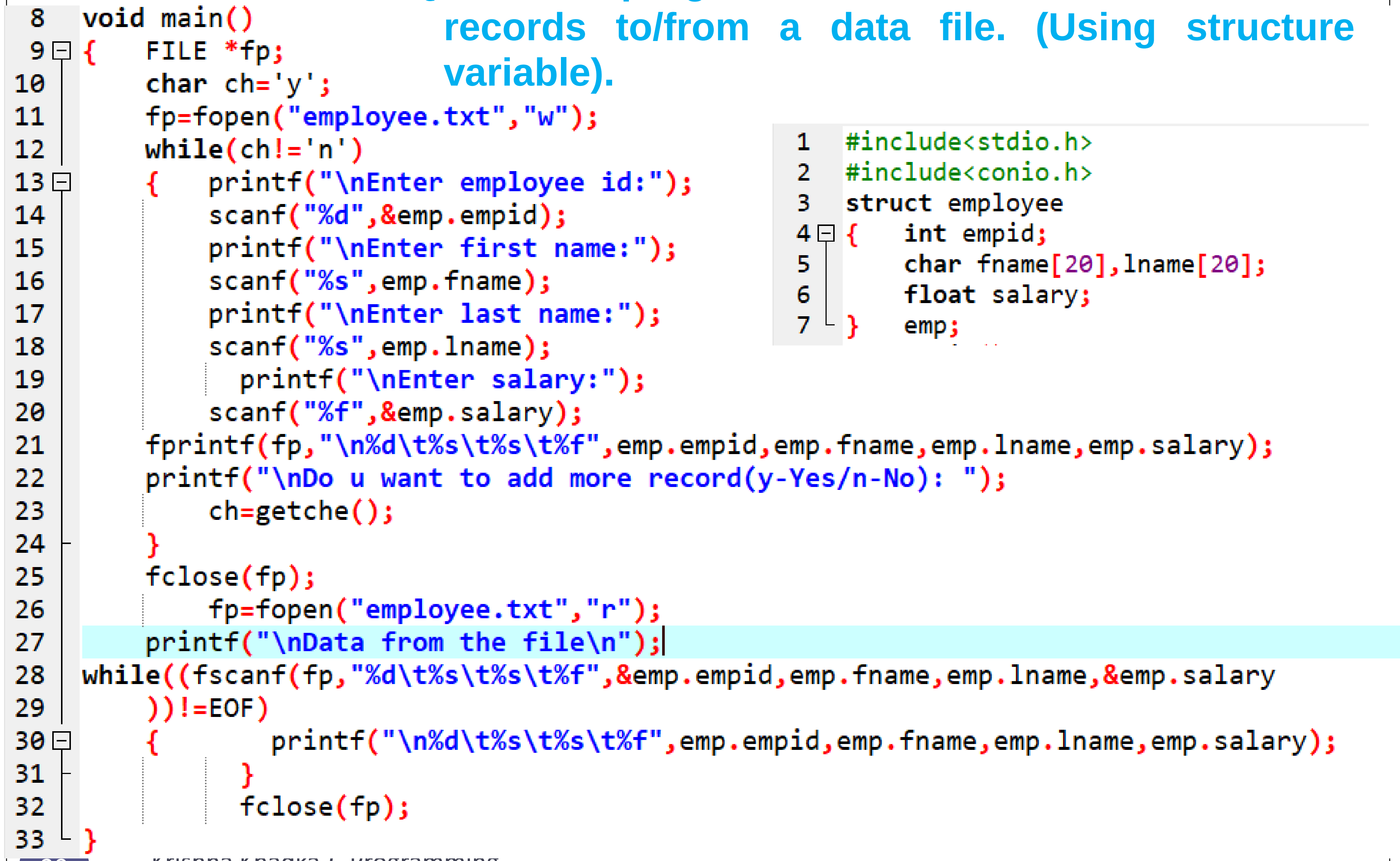
H:\My Drive\Cprogram slides program\file\fileandstructure.exe

```

Enter employee id:1

Enter name:krishna

Enter salary:5000

Data from the file

1       krishna 5000.000000
--------------------------------
Process exited after 13.21 seconds with return value 0
Press any key to continue . . .
```

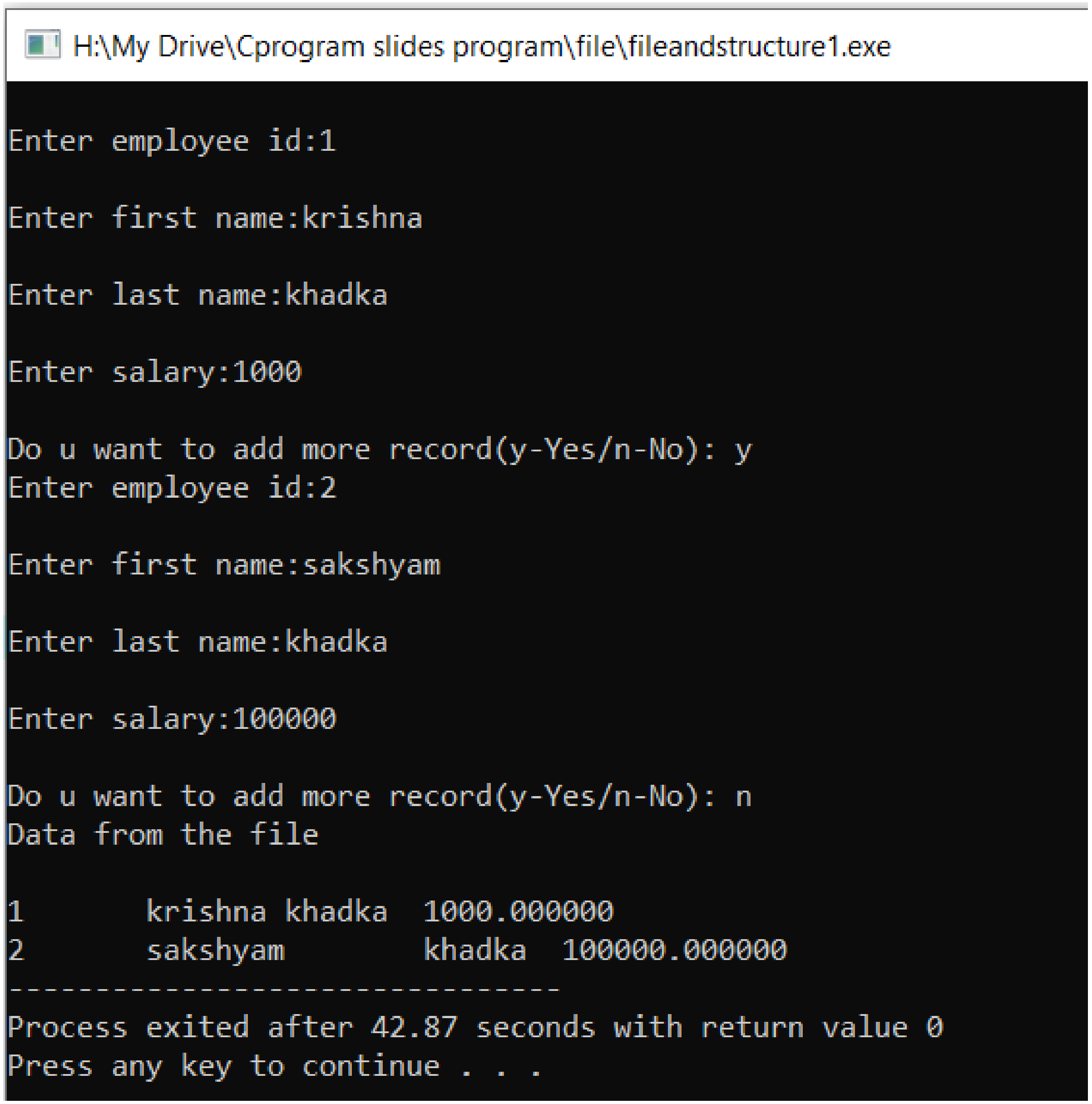
Q. Write a program to write and read successive records to/from a data file. (Using structure variable).



```c
#include<stdio.h>
#include<conio.h>
struct employee
{   int empid;
    char fname[20],lname[20];
    float salary;
}   emp;
```



```c
void main()
{   FILE *fp;
    char ch='y';
    fp=fopen("employee.txt","w");
    while(ch!='n')
    {   printf("\nEnter employee id:");
        scanf("%d",&emp.empid);
        printf("\nEnter first name:");
        scanf("%s",emp.fname);
        printf("\nEnter last name:");
        scanf("%s",emp.lname);
          printf("\nEnter salary:");
        scanf("%f",&emp.salary);
    fprintf(fp,"\n%d\t%s\t%s\t%f",emp.empid,emp.fname,emp.lname,emp.salary);
    printf("\nDo u want to add more record(y-Yes/n-No): ");
        ch=getche();
    }
    fclose(fp);
        fp=fopen("employee.txt","r");
    printf("\nData from the file\n");
while((fscanf(fp,"%d\t%s\t%s\t%f",&emp.empid,emp.fname,emp.lname,&emp.salary
    ))!=EOF)
    {       printf("\n%d\t%s\t%s\t%f",emp.empid,emp.fname,emp.lname,emp.salary);
          }
          fclose(fp);
}
```


H:\My Drive\Cprogram slides program\file\fileandstructure1.exe

```

Enter employee id:1

Enter first name:krishna

Enter last name:khadka

Enter salary:1000

Do u want to add more record(y-Yes/n-No): y
Enter employee id:2

Enter first name:sakshyam

Enter last name:khadka

Enter salary:100000

Do u want to add more record(y-Yes/n-No): n
Data from the file

1       krishna khadka  1000.000000
2       sakshyam        khadka  100000.000000
--------------------------------
Process exited after 42.87 seconds with return value 0
Press any key to continue . . .
```

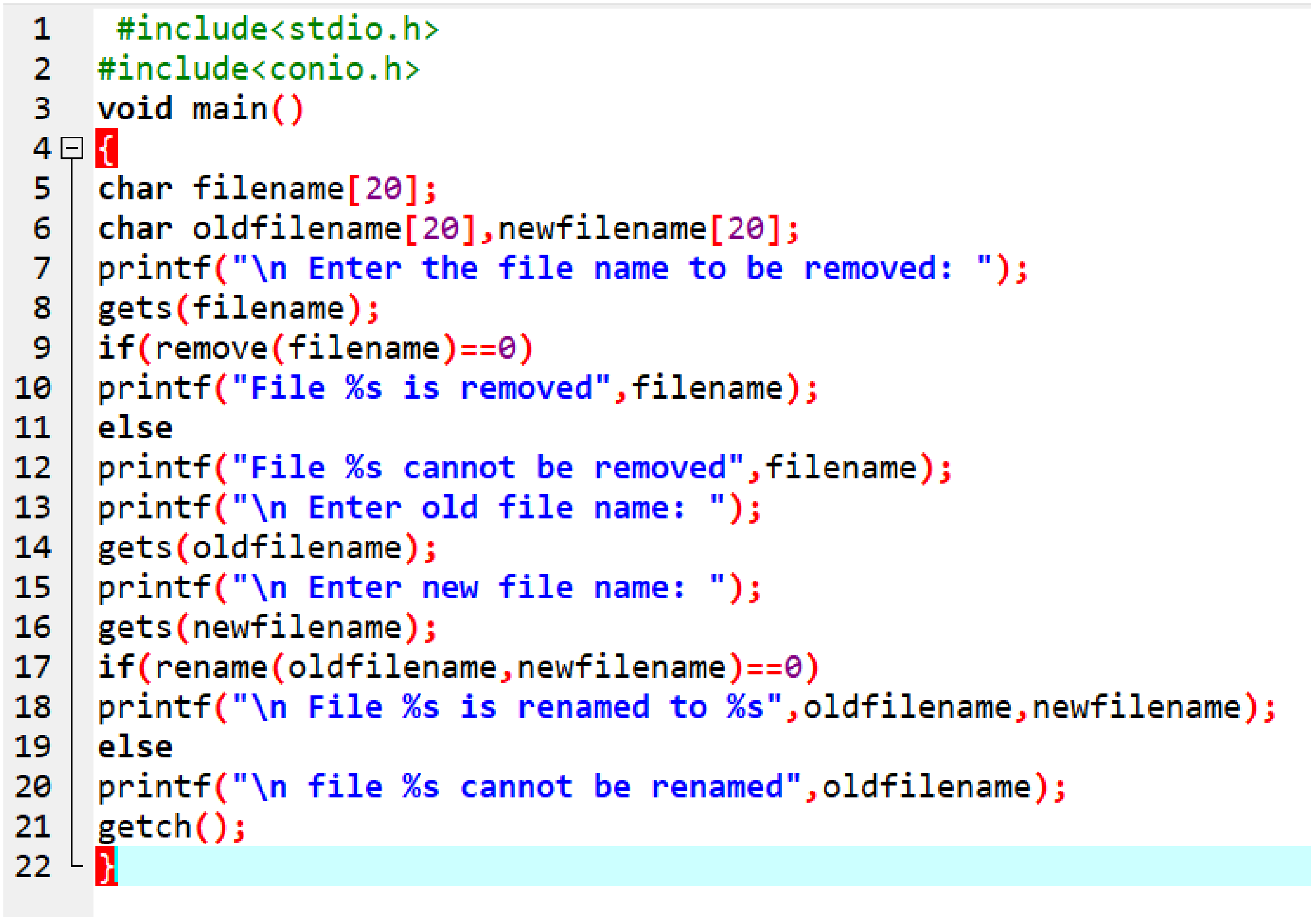
remove and rename functions

- The remove() function helps us to delete the file specified as a parameter and the general syntax is:

```
remove (“filename”);
```

- The rename () function helps us to rename the existing file into new file and the general syntax is:

```
rename (“oldfilename”,“newfilename”);
```



```c
 #include<stdio.h>
#include<conio.h>
void main()
{
char filename[20];
char oldfilename[20],newfilename[20];
printf("\n Enter the file name to be removed: ");
gets(filename);
if(remove(filename)==0)
printf("File %s is removed",filename);
else
printf("File %s cannot be removed",filename);
printf("\n Enter old file name: ");
gets(oldfilename);
printf("\n Enter new file name: ");
gets(newfilename);
if(rename(oldfilename,newfilename)==0)
printf("\n File %s is renamed to %s",oldfilename,newfilename);
else
printf("\n file %s cannot be renamed",oldfilename);
getch();
}
```

Unit 8

Finished