File Handling:

In programming, we may require some specific input data to be generated several numbers of times. Sometimes, it is not enough to only display the data on the console. The data to be displayed may be very large, and only a limited amount of data can be displayed on the console, and since the memory is volatile, it is impossible to recover the programmatically generated data again and again. However, if we need to do so, we may store it onto the local file system which is volatile and can be accessed every time. Here, comes the need of file handling in C. File handling in C enables us to create, update, read, and delete the files stored on the local file system through our C program. The following operations can be performed on a file.

- Creation of the new file
- Opening an existing file
- Reading from the file
- Writing to the file
- Deleting the file

Functions for file handling:

There are many functions in the C library to open, read, write, search and close the file. A list of file functions are given below:

No.	Function	Description
1	fopen()	opens new or existing file
2	fprintf()	write data into the file
3	fscanf()	reads data from the file
4	fputc()	writes a character into the file
5	fgetc()	reads a character from file
6	fclose()	closes the file
7	fseek()	sets the file pointer to given position
8	fputw()	writes an integer to file
9	fgetw()	reads an integer from file
10	ftell()	returns current position
11	rewind()	sets the file pointer to the beginning of the file

Opening File: fopen():

We must open a file before it can be read, write, or update. The fopen() function is used to open a file. The syntax of the fopen() is given below.

1. FILE *fopen(const char * filename, const char * mode);

The fopen() function accepts two parameters:

- The file name (string). If the file is stored at some specific location, then we must mention the path at which the file is stored. For example, a file name can be like "c://some_folder/some_file.ext".
- The mode in which the file is to be opened. It is a string.

We can use one of the following modes in the fopen() function.

Mode	Description
r	opens a text file in read mode
w	opens a text file in write mode
a	opens a text file in append mode
r+	opens a text file in read and write mode
w+	opens a text file in read and write mode
a+	opens a text file in read and write mode
rb	opens a binary file in read mode
wb	opens a binary file in write mode
ab	opens a binary file in append mode
rb+	opens a binary file in read and write mode
wb+	opens a binary file in read and write mode
ab+	opens a binary file in read and write mode

The fopen function works in the following way.

- Firstly, It searches the file to be opened.
- Then, it loads the file from the disk and place it into the buffer. The buffer is used to provide efficiency for the read operations.
- It sets up a character pointer which points to the first character of the file.

Consider the following example which opens a file in write mode.

Closing File: fclose():

The fclose() function is used to close a file. The file must be closed after performing all the operations on it. The syntax of fclose() function is given below:

```
1. int fclose( FILE *fp );
```

C fprintf() and fscanf():

Writing File : fprintf() function:

The fprintf() function is used to write set of characters into file. It sends formatted output to a stream.

Syntax:

```
1. int fprintf(FILE *stream, const char *format [, argument, ...])
```

Example:

```
1. #include <stdio.h>
2. main(){
3.    FILE *fp;
4.    fp = fopen("file.txt", "w");//opening file
5.    fprintf(fp, "Hello file by fprintf...\n");//writing data into file
6.    fclose(fp);//closing file
7. }
```

Reading File : fscanf() function:

The fscanf() function is used to read set of characters from file. It reads a word from the file and returns EOF at the end of file.

Syntax:

```
1. int fscanf(FILE *stream, const char *format [, argument, ...])
```

Example:

```
1. #include <stdio.h>
2. main(){
3.    FILE *fp;
4.    char buff[255];//creating char array to store data of file
5.    fp = fopen("file.txt", "r");
6.    while(fscanf(fp, "%s", buff)!=EOF){
7.    printf("%s ", buff );
8.    }
9.    fclose(fp);
10.}
```

Output:

Hello file by fprintf...

C fputc() and fgetc():

Writing File : fputc() function:

The fputc() function is used to write a single character into file. It outputs a character to a stream.

Syntax:

```
1. int fputc(int c, FILE *stream)
```

```
1. #include <stdio.h>
2. main(){
3.    FILE *fp;
4.    fp = fopen("file1.txt", "w");//opening file
5.    fputc('a',fp);//writing single character into file
6.    fclose(fp);//closing file
7. }
```

Output:

file1.txt

a

Reading File : fgetc() function:

The fgetc() function returns a single character from the file. It gets a character from the stream. It returns EOF at the end of file.

Syntax:

1. int fgetc(FILE *stream)

Example:

```
1. #include<stdio.h>
2. #include<conio.h>
3. void main(){
4. FILE *fp;
5. char c;
6. clrscr();
7. fp=fopen("myfile.txt","r");
8.
9. while((c=fgetc(fp))!=EOF){
10.printf("%c",c);
11.}
12.fclose(fp);
13.getch();
14.}
```

Output:

myfile.txt

this is simple text message

C fseek() function:

The fseek() function is used to set the file pointer to the specified offset. It is used to write data into file at desired location.

Syntax:

```
1. int fseek(FILE *stream, long int offset, int whence)
```

There are 3 constants used in the fseek() function for whence: SEEK_SET, SEEK_CUR and SEEK_END.

Example:

```
1. #include <stdio.h>
2. void main(){
3.     FILE *fp;
4.
5.     fp = fopen("myfile.txt","w+");
6.     fputs("This is javatpoint", fp);
7.
8.     fseek( fp, 7, SEEK_SET );
9.     fputs("sonoo jaiswal", fp);
10.    fclose(fp);
11.}
```

Output:

myfile.txt

This is sonoo jaiswal

C rewind() function:

The rewind() function sets the file pointer at the beginning of the stream. It is useful if you have to use stream many times.

Syntax:

```
1. void rewind(FILE *stream)
```

Example:

File: file.txt

1. this is a simple text

File: rewind.c

```
1. #include<stdio.h>
2. #include<conio.h>
3. void main(){
4. FILE *fp;
5. char c;
6. clrscr();
7. fp=fopen("file.txt","r");
8.
9. while((c=fgetc(fp))!=EOF){
10.printf("%c",c);
11.}
12.
13.rewind(fp);//moves the file pointer at beginning of the file
14.
15.while((c=fgetc(fp))!=EOF){
16.printf("%c",c);
17.}
18.
19.fclose(fp);
20.getch();
21.}
```

Output:

this is a simple textthis is a simple text

As you can see, rewind() function moves the file pointer at beginning of the file that is why "this is simple text" is printed 2 times. If you don't call rewind() function, "this is simple text" will be printed only once.

C ftell() function:

The ftell() function returns the current file position of the specified stream. We can use ftell() function to get the total size of a file after moving file pointer at the end of file. We can use SEEK_END constant to move the file pointer at the end of file.

Syntax:

1. long int ftell(FILE *stream)

Example:

File: ftell.c

```
1. #include <stdio.h>
2. #include <conio.h>
3. void main (){
4.    FILE *fp;
5.    int length;
6.    clrscr();
7.    fp = fopen("file.txt", "r");
8.    fseek(fp, 0, SEEK_END);
9.
10.   length = ftell(fp);
11.
12.   fclose(fp);
13.   printf("Size of file: %d bytes", length);
14.   getch();
15.}
```

Output:

Size of file: 21 bytes