Introduction:

A file is a container in computer storage devices used for storing data.

Why files are needed?

- When a program is terminated, the entire data is lost. Storing in a file will preserve your data even if the program terminates.
- If you have to enter a large number of data, it will take a lot of time to enter them all.
- However, if you have a file containing all the data, you can easily access the contents of the file using a few commands in C.
- You can easily move your data from one computer to another without any changes.

Types of Files

When dealing with files, there are two types of files you should know about:

1. Text files
2. Binary files

1. Text files

Text files are the normal **.txt** files. You can easily create text files using any simple text editors such as Notepad.

When you open those files, you'll see all the contents within the file as plain text. You can easily edit or delete the contents.

They take minimum effort to maintain, are easily readable, and provide the least security and takes bigger storage space.

2. Binary files

Binary files are mostly the **.bin** files in your computer.

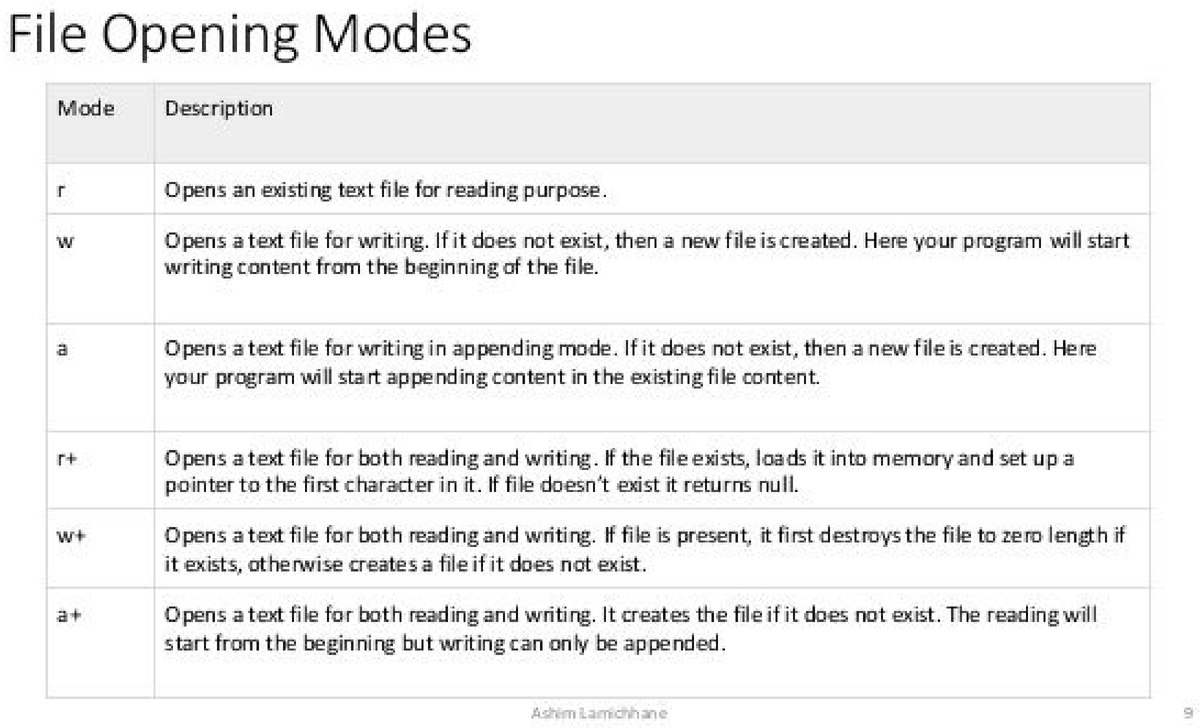
Instead of storing data in plain text, they store it in the binary form (0's and 1's).

They can hold a higher amount of data, are not readable easily, and provides better security than text files.

File Operations

In C, you can perform four major operations on files, either text or binary:

1. Creating a new file
2. Opening an existing file
3. Closing a file
4. Reading from and writing information to a file



File Opening Modes

Mode	Description
r	Opens an existing text file for reading purpose.
w	Opens a text file for writing. If it does not exist, then a new file is created. Here your program will start writing content from the beginning of the file.
a	Opens a text file for writing in appending mode. If it does not exist, then a new file is created. Here your program will start appending content in the existing file content.
r+	Opens a text file for both reading and writing. If the file exists, loads it into memory and set up a pointer to the first character in it. If file doesn't exist it returns null.
w+	Opens a text file for both reading and writing. If file is present, it first destroys the file to zero length if it exists, otherwise creates a file if it does not exist.
a+	Opens a text file for both reading and writing. It creates the file if it does not exist. The reading will start from the beginning but writing can only be appended.

Ashim Lamichhane 9

Working with files

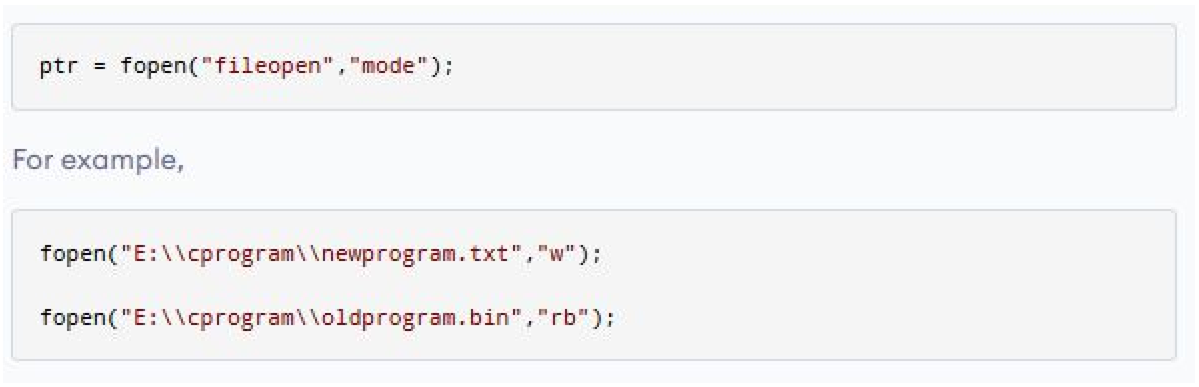
When working with files, you need to declare a pointer of type file. This declaration is needed for communication between the file and the program.

```
FILE *fptr;
```

Opening or Creating a file:

Opening a file is performed using the fopen() function defined in the stdio.h header file.

The syntax for opening a file in standard I/O is:



```
ptr = fopen("fileopen","mode");
```

For example,

```
fopen("E:\\cprogram\\newprogram.txt","w");

fopen("E:\\cprogram\\oldprogram.bin","rb");
```

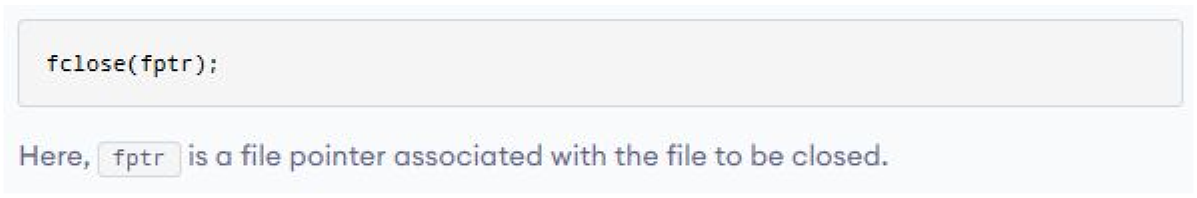
- Let's suppose the file newprogram.txt doesn't exist in the given location. The first function creates a new file named newprogram.txt and opens it for writing as per the mode **'w'**.The writing mode allows you to create and edit (overwrite) the contents of the file.

- Now let's suppose the second binary file oldprogram.bin exists in the given location. The second function opens the existing file for reading in binary mode **'rb'**.

Closing a file

The file (both text and binary) should be closed after reading/writing.

Closing a file is performed using the fclose() function.



```
fclose(fptr);
```

Here, `fptr` is a file pointer associated with the file to be closed.

Reading and writing to a text file

For reading and writing to a text file, we use the functions fprintf() and fscanf().

They are just the file versions of printf() and scanf(). The only difference is that fprint() and fscanf() expects a pointer to the structure FILE.

Example: Writing to a text file

```
#include<stdio.h>
void main()
{
   FILE *fptr;
   char description[100];
   int number;
   fptr=fopen("D:\demo.txt","w");
   if(fptr==NULL)
   {
         printf("File cannot be created.");
         exit(1);
   }

   printf("Write desctiption to store in File:");
   fgets(description,100,stdin);
   printf("Enter your phone no. to store in File:");
   scanf("%d",&number);
   fprintf(fptr,"%s",description);
   fprintf(fptr,"%d",number);
   printf("File saved successfully!");
   fclose(fptr);

}
```

Example: Reading from a text file

```
//Example of writing to a text file

#include<stdio.h>
void main()
{
    FILE *fptr;
    char description[100];
    int number;
    fptr=fopen("D:\demo.txt","r");
    if(fptr==NULL)
    {
            printf("File cannot be opened.");
            exit(1);
    }

    while (fgets(description, 100, fptr) != NULL)
           printf("%s", description);

    fclose(fptr);

}
```