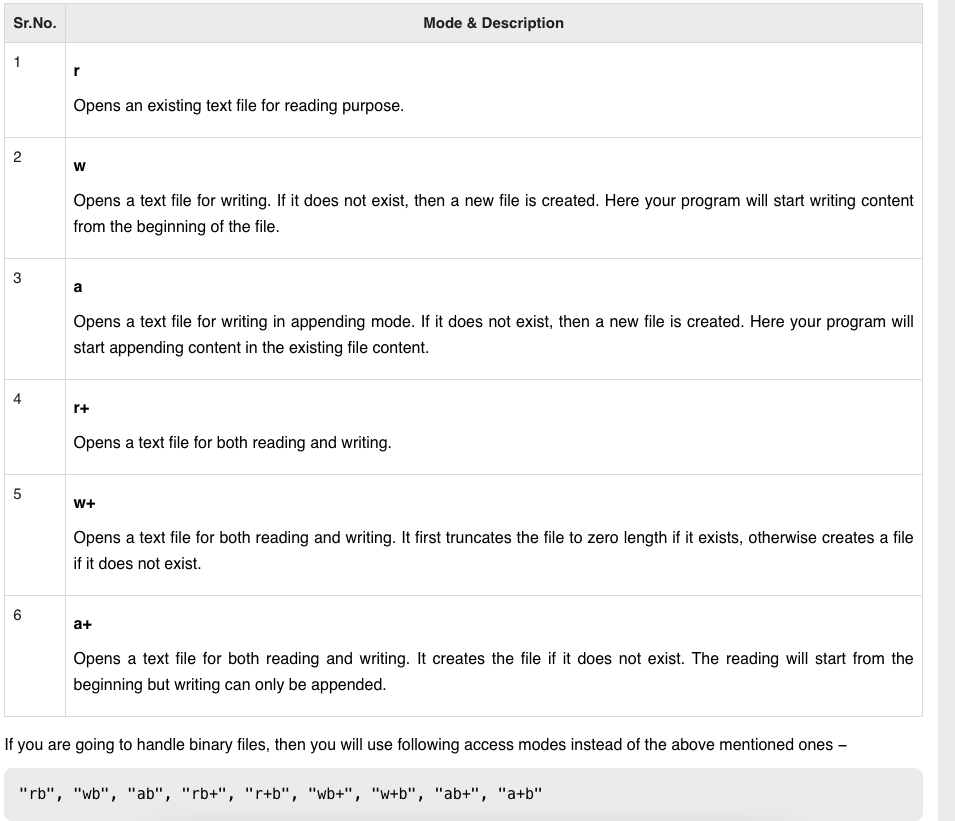
# **C - File I/O**

Opening Files

You can use the **fopen( )** function to create a new file or to open an existing file. This call will initialize an object of the type **FILE**, which contains all the information necessary to control the stream. The prototype of this function call is as follows −

FILE \*fopen( const char \* filename, const char \* mode );

Here, **filename** is a string literal, which you will use to name your file, and access **mode** can have one of the following values –



## Closing a File

To close a file, use the fclose( ) function. The prototype of this function is −

int fclose( FILE \*fp );

The **fclose(-)** function returns zero on success, or **EOF** if there is an error in closing the file. This function actually flushes any data still pending in the buffer to the file, closes the file, and releases any memory used for the file. The EOF is a constant defined in the header file **stdio.h**.

There are various functions provided by C standard library to read and write a file, character by character, or in the form of a fixed length string.

## Writing a File

Following is the simplest function to write individual characters to a stream −

int fputc( int c, FILE \*fp );

The function **fputc()** writes the character value of the argument c to the output stream referenced by fp. It returns the written character written on success otherwise **EOF** if there is an error. You can use the following functions to write a null-terminated string to a stream −

int fputs( const char \*s, FILE \*fp );

The function **fputs()** writes the string **s** to the output stream referenced by fp. It returns a non-negative value on success, otherwise **EOF** is returned in case of any error. You can use **int fprintf(FILE \*fp,const char \*format, ...)** function as well to write a string into a file. Try the following example.

Make sure you have **/tmp** directory available. If it is not, then before proceeding, you must create this directory on your machine.

#include <stdio.h>

main() {

FILE \*fp;

fp = fopen("/tmp/test.txt", "w+");

fprintf(fp, "This is testing for fprintf...\n");

fputs("This is testing for fputs...\n", fp);

fclose(fp);

}

When the above code is compiled and executed, it creates a new file **test.txt** in /tmp directory and writes two lines using two different functions. Let us read this file in the next section.