Unit : 7

File Handling in c

- A file represents a sequence of bytes, regardless of it being a text file or a binary file. C programming language provides access on high level functions as well as low level (OS level) calls to handle file on your storage devices. This chapter will take you through the important calls for file management.

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.

- 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.
- w+ Opens a text file for both reading and writing. It first truncates 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.

Closing a File

- To close a file, use the fclose( ) function.

  int fclose(FILE *fp);

Writing a File

- The function **fputc**() writes the character value of the argument c to the output stream referenced by fp.

- The function **fputs**() writes the string **s** to the output stream referenced by fp. It returns a non-negative value on success

Reading a File

- The **fgetc()** function reads a character from the input file referenced by fp. The return value is the character read, or in case of any error, it returns **EOF**.

- The functions **fgets()** reads up to n-1 characters from the input stream referenced by fp. It copies the read string into the buffer **buf**, appending a **null** character to terminate the string.