CSE 06131223 ♦ CSE 06131224

Structured Programming

Lecture 23

File Management in C (1)

Prepared by

Md. Mijanur Rahman, Prof. Dr.
Dept. of Computer Science and Engineering
Jatiya Kabi Kazi Nazrul Islam University, Bangladesh
www.mijanrahman.com

Contents

File Management in C

Introduction

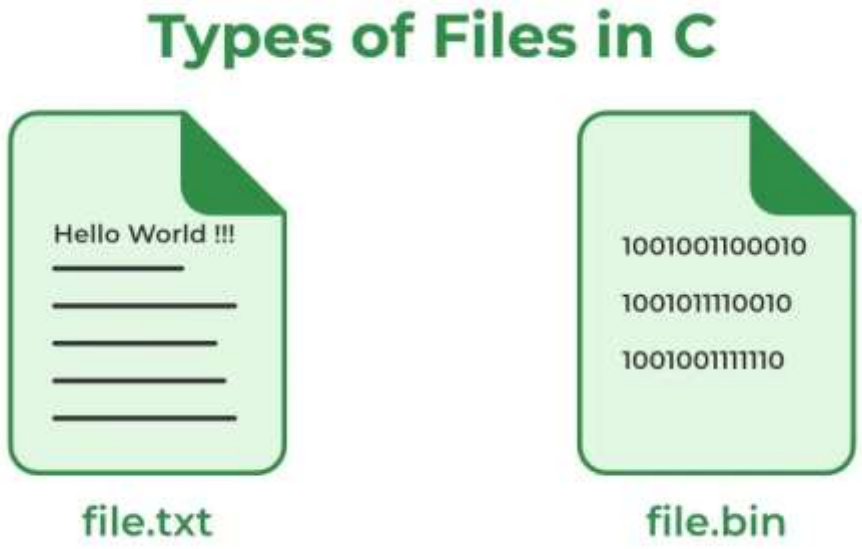
- File handing in C is the process in which we create, open, read, write, and close operations on a file.
- C language provides different functions such as fopen(), fwrite(), fread(), fseek(), fprintf(), etc. to perform input, output, and many different C file operations in our program.
- So far the operations using the C program are done on a prompt/terminal which is not stored anywhere. The output is deleted when the program is closed.
- But in the software industry, most programs are written to store the information fetched from the program. The use of file handling is exactly what the situation calls for.

Why do we need File Handling in C?

- In order to understand why file handling is important, let us look at a few features of using files:
  - **Reusability:** The data stored in the file can be accessed, updated, and deleted anywhere and anytime providing high reusability.
  - **Portability:** Without losing any data, files can be transferred to another in the computer system. The risk of flawed coding is minimized with this feature.
  - **Efficient:** A large amount of input may be required for some programs. File handling allows you to easily access a part of a file using few instructions which saves a lot of time and reduces the chance of errors.
  - **Storage Capacity:** Files allow you to store a large amount of data without having to worry about storing everything simultaneously in a program.

Types of Files in C

- A file can be classified into two types based on the way the file stores the data. They are as follows:
- **Text Files**
- **Binary Files**



Types of Files in C

- **Text Files:** A text file contains data in the form of ASCII characters and is generally used to store a stream of characters.
  - Each line in a text file ends with a new line character (‘\n’).
  - It can be read or written by any text editor.
  - They are generally stored with .txt file extension.
  - Text files can also be used to store the source code.

file.txt

Types of Files in C

- **Binary Files:** A binary file contains data in binary form (i.e. 0's and 1's) instead of ASCII characters. They contain data that is stored in a similar manner to how it is stored in the main memory.
  - The binary files can be created only from within a program and their contents can only be read by a program.
  - More secure as they are not easily readable.
  - They are generally stored with .bin file extension.

C File Operations

- C file operations refer to the different possible operations that we can perform on a file in C such as:
  - Creating a new file – fopen() with attributes as “a” or “a+” or “w” or “w+”
  - Opening an existing file – fopen()
  - Reading from file – fscanf() or fgets()
  - Writing to a file – fprintf() or fputs()
  - Moving to a specific location in a file – fseek(), rewind()
  - Closing a file – fclose()

File Pointer in C

- A file pointer is a reference to a particular position in the opened file. It is used in file handling to perform all file operations such as read, write, close, etc.
- We use the **FILE** macro to declare the file pointer variable. The **FILE** macro is defined inside <stdio.h> header file.

- **Syntax of File Pointer**

```
FILE* pointer_name;
```

- File Pointer is used in almost all the file operations in C.

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

Defining and Opening a File

- 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.

```
FILE* fopen(const char *file_name, const char *access_mode);
```

- **Parameters:**
  - file_name: name of the file when present in the same directory as the source file. Otherwise, full path.
  - access_mode: Specifies for what operation the file is being opened.
- **Return Value:**
  - If the file is opened successfully, returns a file pointer to it.
  - If the file is not opened, then returns NULL.

Defining and Opening a File

- **File opening modes in C:** File opening modes or access modes specify the allowed operations on the file to be opened. They are passed as an argument to the fopen() function.

- Some of the commonly used file access modes are listed here:

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

Defining and Opening a File

- 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.

Defining and Opening a File

- **Example of Opening a File:**

```c
// C Program to illustrate file opening
#include <stdio.h>
#include <stdlib.h>

int main()
{
    // file pointer variable to store the value returned by
    // fopen
    FILE* fptr;

    // opening the file in read mode
    fptr = fopen("filename.txt", "r");

    // checking if the file is opened successfully
    if (fptr == NULL) {
        printf("The file is not opened. The program will "
               "now exit.");
        exit(0);
    }

    return 0;
}
```

Output

```
The file is not opened. The program will now exit.
```

Creating a File in C

- The **fopen()** function can not only open a file but also can create a file if it does not exist already.
- For that, we have to use the modes that allow the creation of a file if not found such as w, w+, wb, wb+, a, a+, ab, and ab+.

- The syntax for creating a file in C:

```
FILE *fptr;
fptr = fopen("filename.txt", "w");
```

Creating a File in C

- Example of Creating a File:

```c
// C Program to create a file
#include <stdio.h>
#include <stdlib.h>

int main()
{
    // file pointer
    FILE* fptr;

    // creating file using fopen() access mode "w"
    fptr = fopen("file.txt", "w");

    // checking if the file is created
    if (fptr == NULL) {
        printf("The file is not opened. The program will "
               "exit now");
        exit(0);
    }
    else {
        printf("The file is created Successfully.");
    }

    return 0;
}
```

Output

```
The file is created Successfully.
```

Closing a File

- The **fclose()** function is used to close the file. After successful file operations, you must always close a file to remove it from the memory.
- **Syntax of fclose():**

```
fclose(file_pointer);
```

- where the file_pointer is the pointer to the opened file.
- For Example:

```
FILE *fptr ;
fptr= fopen("fileName.txt", "w");
---------- Some file Operations -------
fclose(fptr);
```

THE END