UNIT V FILE PROCESSING 9

Files – Types of file processing: Sequential access, Random access – Sequential access file - Random access file - Command line arguments.

FILES

A file represents a sequence of bytes on the disk where a group of related data is stored. File is created for permanent storage of data. It is a readymade structure.

Why are files 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 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 that you can easily create using Notepad or any simple text editors. 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 least security and take up a 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 provide better security than text files.

File Operations

In C, you can perform four major operations on the file, 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
5. C provides a number of functions that help to perform basic file operations.

Following are the functions,

1. fopen() - create a new file or open a existing file
2. fclose() - closes a file
3. getc() - reads a character from a file
4. putc() - writes a character to a file
5. fscanf() - reads a set of data from a file
6. fprintf() - writes a set of data to a file
7. getw() - reads a integer from a file
8. putw() - writes a integer to a file
9. fseek() - set the position to desire point
10. ftell() - gives current position in the file
11. rewind() - set the position to the beginning point

Opening a File or Creating a File

The fopen() function is used to create a new file or to open an existing file.

Syntax:

***fp = FILE *fopen(const char *filename, const char *mode);**

Here, *fp is the FILE pointer (FILE *fp), which will hold the reference to the opened (or created) file.

filename is the name of the file to be opened and mode specifies the purpose of opening the file.

Mode can be of following types,

1. r - opens a text file in reading mode
2. w - opens or create a text file in writing mode.
3. a - opens a text file in append mode
4. r+ - opens a text file in both reading and writing mode
5. w+ - opens a text file in both reading and writing mode
6. a+ - opens a text file in both reading and writing mode
7. rb - opens a binary file in reading mode
8. wb - opens or create a binary file in writing mode
9. ab - opens a binary file in append mode
10. rb+ - opens a binary file in both reading and writing mode
11. wb+ - opens a binary file in both reading and writing mode
12. ab+ - opens a binary file in both reading and writing mode

Closing a File

The fclose() function is used to close an already opened file.

Syntax :

int fclose(FILE *fp);

Here fclose() function closes the file and returns zero on success, or EOF if there is an error in closing the file. This EOF is a constant defined in the header file stdio.h.
