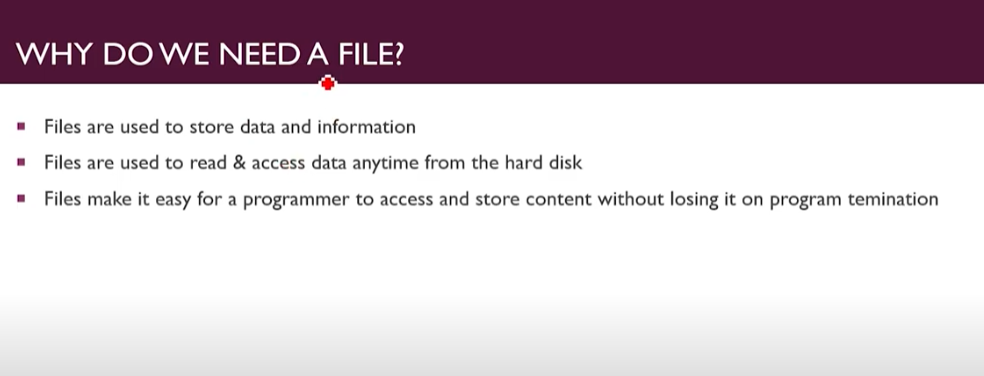
Content 45

File I/O In C:



**Data:-** Data are bits of Storage which actually collected by us.

**Information:-** andWhen do process on data and convert it into a meaning full thing then it becomes information.

we will be moving towards **files in C**. Files are an essential part of any program as we can take input from and print output in files. We can also save a lot of program space by accessing the file's data only when needed, making the program more efficient and faster.

#### Purpose of files in C:

* Files are used to store content hence **reducing the program's size**.
* We can **read or access data**from files.
* The data in files remain stored even after the program's execution is **terminated**.

Files are stored in **non-volatile memory**. To understand what a non-volatile memory is, we have to see the difference between volatile and non-volatile memory.

|  |  |
| --- | --- |
| **VOLATILE MEMORY** | **NON- VOLATILE MEMORY** |
| The data can only remain in it while the computer’s power is on. | The data will also be present in it while the computer’s power is off. |
| Can only hold information when having a constant power supply | Can also hold information, in case of inconstant power supply |
| Will hold data for a short period. | Will hold data for a long term. |
| RAM is an example | Hard Disk is an example. |

So due to the above differences, we can conclude the reasons we need files.

#### Types of Files:

There are two types of files:

* Binary Files
* Text Files

#### Binary Files:

* Binary files stores data in 01 i.e., **binary format**.
* They are not directly readable.
* An application or software is required to read binary files.
* An example is a **.doc** file.

#### Text Files:

* They store data in simple **text format**.
* They are directly readable.
* No software is required to access them.
* An example is a**.txt** file.

#### Operations on files:

By using C language, we can perform four different tasks related to files. We will see them theoretically in this tutorial, and in the next one's, we will see their practical implementation.

#### Creating a File:

We can create a file using C language, in any directory, without even leaving our compiler. We can select the name or type we want our file to have, along with its location.

#### Opening a File:

We can open an existing file and create a new file and open it using our program. We can perform different operations on a file after it has been opened.

#### Closing a File:

When we are done with the file, meaning that we have performed whatever we want to perform on our file, we can close the file using the close function.

#### Read/Write to a file:

After opening a file, we can access its contents and read, write, or update them.

#### Conclusion:

We can perform many operations using files, as we can read, write, or append them. We can also open, close, or create a file. Using files in our programs can make them much more efficient and save a lot of memory space. We can also store our data for an extended period.

