C++ File I/O

The file is a patent of data stored on the disk. Anything written inside the file is called a patent, for example: #include is a patent. The text file is the combination of multiple types of characters, for example, the semicolon ; is a character.

The computer reads these characters in the file with the help of the ASCII code. Every character is mapped to some decimal number. For example, the ASCII code for the character A is 65, which is a decimal number. These decimal numbers are converted into binary numbers to make them readable for the computer because the computer can only understand the language of 0 & 1.

A large program deployed for heavy applications cannot function without files, since we can get input from them as well as print output from them very easily. 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.

Files are stored in non-volatile memory, which is better in terms of storing data. Non-volatile memory stays intact even after the system shuts down. They get retrieved again after the system is turned on.

Operations on files

There are basically four operations we can perform on files in C.

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.

File I/O Functions

These are some useful classes for working with files in C++:

- fstream - A combination of ofstream and ifstream: creates, reads, and writes to files
- ofstream - creates and writes to files
- ifstream - reads from files

The fstream library allows us to handle files. So, to be able to use files in a program, one must include the <fstream> header file.

A. Opening a file

In order to work with files in C++, we will first have to open it. Primarily, there are two ways to open a file:

- Using the constructor
- Using the member function open() of the class

A file could be opened for a number of uses. It could be to write to it or to read from it.

Consider an example that demonstrates the opening of a file using a constructor.

```
#include <iostream>

#include <fstream>


using namespace std;


int main()

{

   ofstream out("example.txt");

   return 0;

}
```

The file example.txt gets created if it's not already there in the system and opened. Object out gets created of the type ofstream, which means it could only be used to write into the opened file.

This is how we use the constructor ofstream to open a file. Another example demonstrates the use of the ifstream constructor.

```
#include <iostream>

#include <fstream>


using namespace std;


int main()

{

   ifstream in("example.txt");

   return 0;

}
```

For this to work, a file named example.txt must already be created and present in the same folder as that of the program. Object in gets created of the type ifstream, which means it could only be used to read from the file.

B. Closing a file

When working with C++, closing open files is considered a good practice. A programmer often makes the mistake of not closing an open file. This becomes crucial because files do not automatically get closed after a program uses them. The closing has to be done manually.

To close a file, we have to use the close method. This is how we use them in C++.

Syntax:

```
file_objectname.close();
```

C. Writing to a file

Writing to a file is as easy as printing any other stuff in C++. It is very similar to what we used to do when we had to print an output in the terminal. In order to write to a file, we use the insertion operator (<<). First, we create an object of the type ofstream and pass the name of the file along with its extension to the method. And then, use the insertion operator to write stuff in the file fed to the object.

Consider an example demonstrating how we write to a file.

Example:

```
#include <iostream>
#include <fstream>

using namespace std;

int main()
{
   string str = "Welcome_To_CodeWithHarry!";
   ofstream out("example.txt");
   out << str;
   return 0;
}
```

Output in the example.txt file:

Welcome_To_CodeWithHarry!

D. Reading a file

Reading from a file is as easy as reading any other stuff as an input in C++. It is very similar to what we used to do when we had to read input from the terminal. In order to read from a file, we use the extraction operator (>>). First, we create an object of the type ifstream and pass the name of the file along with its extension to the method. And then, use the extraction operator to read stuff from the file fed to the object.

Consider an example demonstrating how we read from a file.

Example:

```
#include <iostream>
#include <fstream>

using namespace std;

int main()
{
```

```
    string str;
    ifstream in("example.txt");
    in >> str;
    cout << str;
    return 0;
}
```

The file example.txt had “Welcome_To_CodeWithHarry!” as its content, hence the output:

Welcome_To_CodeWithHarry!