**File I/O in C++: Working with Files :**

The file is a patent of data which is stored in 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, semicolon “;” is a character.

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

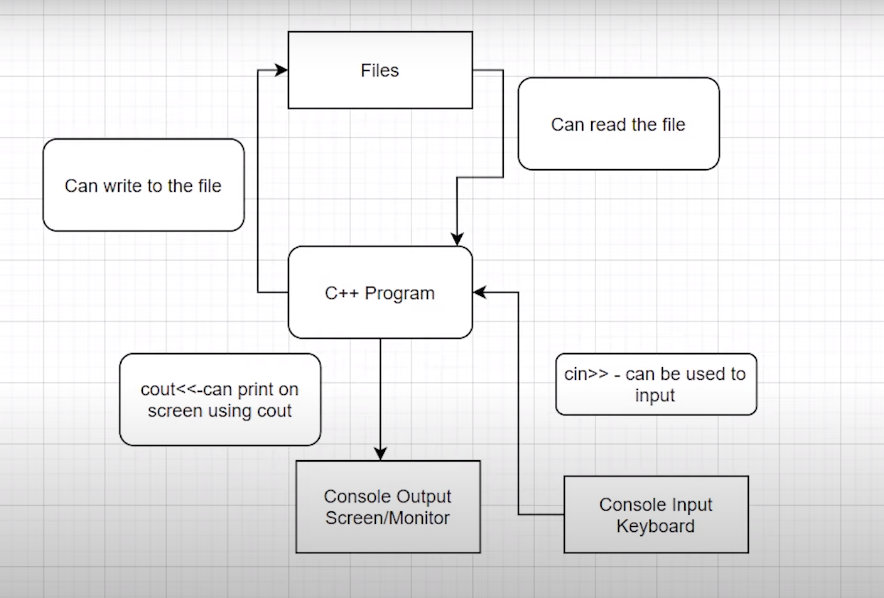
The reason that computers can only understand binary numbers is that a computer is made up of switches and switches only perform two operations “true” or “false”.

**File Input and Output in C++:**

The file can be of any type whether it is a file of a C++ program, file of a game, or any other type of file. There are two main operations which can be performed on files

* **Read File**
* **Write File**

An image is shown below to show the process of file read and write.



***Figure 1: File Read and Write Diagram***

As shown in figure 1,

1. The user can provide input to the C++ program by using keyboard through “cin>>” keyword
2. The user can get output from the C++ program on the monitor through “cout<<” keyword
3. The user can write on the file
4. The user can read the file

**File I/O in C++: Reading and Writing Files:**

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

* fstreambase
* ifstream --> derived from fstreambase
* ofstream --> derived from fstreambase

In order to work with files in C++, you will have to open it. Primarily, there are 2 ways to open a file:

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

### File I/O in C++: Read/Write in the Same Program & Closing Files:

In this tutorial, we’ll learn about creating a program that will read from a file and write to the file in the same program using a constructor.

Before jumping on to the main thing, we’ll first give ourselves a quick revision of the things we had learned previously.

We had learned about the three most useful classes when we talk about File I/O, namely,

1. fstreambase
2. ifstream
3. ofstream.

All the above three classes can be used in a program by first including the header file, fstream.

**Reading File Operation Output:**

We learnt reading from a file using ifstream. Below snippet will help you recollect the same.

string st;

// Opening files using constructor and reading it

ifstream in("this.txt"); // Read operation

in>>st;

**Writing File Operation Output:**

We learnt reading from a file using ofstream. Below snippet will help you recollect the same.

string st = "Harry bhai";

// Opening files using constructor and writing it

ofstream out("this.txt"); // Write operation

out<<st;

Let me make these codes functional in the same program for you to easily understand the workflow.

Suppose we have a file named sample60.txt in the same directory, we can easily call the file infinite number of times in the same program only by maintaining different connections for different purposes, using

<object\_name>.close();

Now, let’s move on to our systems. Open your editors as well. Don’t forget to include the header file, <fstream>.

Follow these steps below to first write into the empty file:

1. Create a text file “sample60.txt” in the same directory as that of the program.
2. Create a string variable name.
3. Create an object hout(name it whatever you wish) using ofstream passing the text file, sample60.txt into it. This establishes a connection between your program and the text file.
4. Take input from the user using cin into the name string.(You can write manually as well)
5. Pass this name string to the object hout. The string name gets written in the text  file.
6. Disconnect the file with the program since we are done writing to it using hout.close().

Since the file has been disconnected from the program, we can connect it again for any other purpose in the same program independently.

Follow these steps below to read from the file we just wrote into:

1. Create a string variable content.
2. Create an object hin(name it whatever you wish) using ifstream passing the text file, sample60.txt into it. This establishes a new connection between your program and the text file.
3. Fill in the string using the object hin. (Use getline, which we talked about in the last video, to take into input the whole line from the text file.)
4. Give output to the user, the string we filled in with the content in the text file.
5. Disconnect the file with the program since we are done reading from it using hin.close().

#include<iostream>

#include<fstream>

using namespace std;

int main(){

// connecting our file with hout stream

ofstream hout("sample60.txt");

// creating a name string variable and filling it with string entered by the user

string name;

cout<<"Enter your name: ";

cin>>name;

// writing a string to the file

hout<<name + " is my name";

// disconnecting our file

hout.close();

// connecting our file with hin stream

ifstream hin("sample60.txt");

// creating a content string variable and filling it with string present there in the text file

string content;

hin>>content;

cout<<"The content of the file is: "<<content;

// disconnecting our file

hin.close();

return 0;

}

Let’s run the program we just created, The output will look like this:

Enter your name: Harry

The content of the file is: Harry

So when we input a string “Harry” into the text file, it gets written there in the file as below, and when we read it from the file, it gives output as below. Since we used hin and not getline, it could read just the first word.

The content of the file is: Harry

**File I/O in C++: open() and eof() functions:**

In this tutorial, we are going to learn about the member functions open and eof of the objects we learnt about previously.

I remember teaching you all about the two methods to open a text file in our C++ program, first one using a constructor which we discussed in the last tutorial, and the second one, using the member function open, which is to be dealt with today.

**Using the member function open:**

The member function open is used to connect the text file to the C++ program when passed into it.

Understanding the snippet below:

1. Unlike what we did earlier passing the text file in the object while creating it, we’ll first just declare an object out(any name you wish) of the type ofstream and use its open method to open the text file in the program.
2. We’ll pass some string lines to the text file using the out operation.
3. We’ll now close the file using the close function. Now closing is explicitly used to make the system know that we are done with the file. It is always good to use this.

This was all about writing to a file. We’ll now move to the eof function’s vitality in File I/O.

#include <iostream>

#include <fstream>

using namespace std;

int main()

{

// declaring an object of the type ofstream

ofstream out;

//connecting the object out to the text file using the member function open()

out.open("sample60.txt");

//writing to the file

out <<"This is me\n";

out <<"This is also me";

//closing the file connection

out.close();

return 0;

}

Copy

**Using the member function eof:**

The member function eof(End-of-file) returns a boolean true if the file reaches the end of it and false if not.

Understanding the snippet below:

1. We’ll first declare an object in(any name you wish) of the type ifstream and use its open method similar to what we did above, to open the text file in the program.
2. And now, we’ll declare the string variable st to store the content we’ll receive from the text file sample60.txt.
3. Now since we not only want the first or some two or three strings present in          the text file, but the whole of it, and we have no idea of what the length of the file is, we’ll use a while loop.
4. We’ll run the while loop until the file reaches the end of it, and that gets checked by using eof() , which returns 1 or true if the file reaches the end. Till then a 0 or false.
5. We’ll use getline to store the whole line in the string variable st. Don’t forget to include the header file <string>.
6. This program now successfully prints the whole content of the text file.