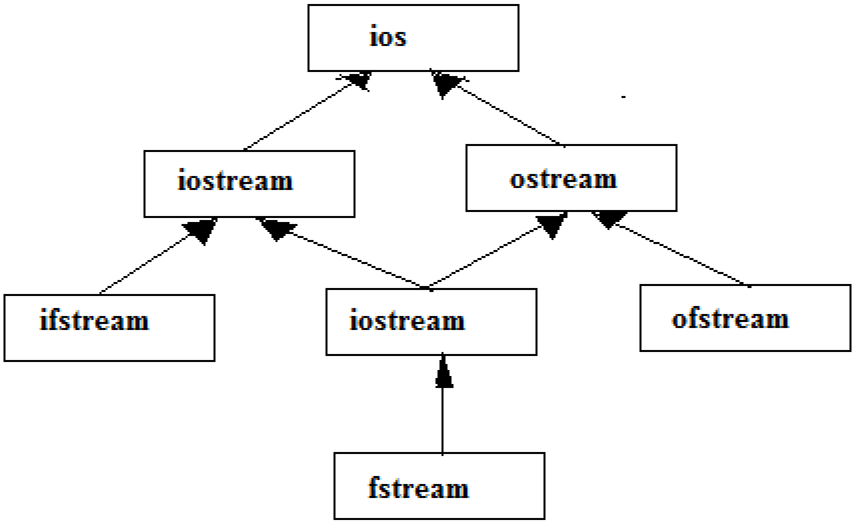
FILE HANDLING in C++

Till now you have been watching the output of a c++ program through iostream on the monitor of your computer. You were required to execute the program every time to see the output, and you somehow wanted to save the output in a file, but iostream class does not permit that operation. If we wish to operate on file, save some user content which he/she enters through the the keyboard, then we need the help of another stream called fstream class as a header in your program.

A file stream represents a sequence of characters flowing to or from a program where the source or the destination is a file. C++ provides all object-oriented stream facilities through its stream library. A file stream connects to a program through a file object. The object has a state and member functions that manages access to the stream. These functions enable access to any byte within the stream.

The C++ stream library is founded upon the ios base class. The input (istream) and output (ostream) classes are derived from this base class. The two classes designed for file streams are derived from these two classes: the file input class (ifstream) from the istream class and the file output class (ofstream) from the ostream class. A third class called fstream holds the information for both input and output file streams and is derived from the iostream class, which is derived from the istream and ostream classes. The fstream header file contains the declarations for the three file stream
classes



Modes of Files:

Mode of a file specify in which format the data will be written in the file. These are two mode in which data can be written to a file and hence accordingly, file mode are:

a. Text files
b. Binary files

These are not the type of file, let us understand the difference between the two modes:

	Text File Mode	Binary Files Mode
Conversion base-2 to base-10	Binary data(base 2) is stored in base 10 format	Binary data is stored in binary format
translation	in text files various character translations are performed such as “\r+\f” is converted into “\n”	in binary files no such translations are performed
Size	Let us take integer variable example, its values is always stored in four bytes in memory, When it is output to disk file in “text mode”. If its value is ‘1’, it will occupy one byte in output file although it occupies four byte in memory. Similarly, If its value is ‘12’, it will occupy two bytes in disk output file, if its value is ‘123’ it will occupy three bytes byte in output file although they will occupy four byte in memory.	When the value of nueric variable is output in binary mode, it occupies the same number of bytes in output file as it does in memory. Thus, if number of such valuess are output in binary mode, the size of output file will always be multiple of four. Obviously the size of file and divide it by four to easily determine number of integers(records) that are stored in the file

Opening a file	ofstream obj (“myfile.txt”);
	or
	ofstream obj;
	obj.open(“myfile.txt”);
		ofstream obj (“myfile.txt”,ios::binary);
		or
		ofstream obj;
		obj.open(“myfile.txt”, ios::binary);

Appending a file	ofstream obj(“myfile.txt”,ios::app);
	or
	ofstream obj;
	obj.open(“myfile.txt”, ios::app);
		ofstream obj (“myfile.txt”,ios::app|ios::binary);
		or
		ofstream obj;
		obj.open(“myfile.txt”, ios::app | ios::binary);

Reading from file	ifstream obj (“myfile.txt”);
	or
	ifstream obj ;
	obj.open(“myfile.txt”);
		ifstream obj (“myfile.txt”, ios::binary);
		or
		ifstream obj ;
		obj.open(“myfile.txt”, ios::binary);

Open() and close() methods

We use the methods open() and close() to open a file for some operations such as read, write, append etc and to close an opened file. The following code illustrate their use and shows how to write some contents entered through the keyboard and save (write) it to a file ABC.txt. The program creates an object of the ofstream class. The object then can access the open() and close() methods.

Data can be written into a file using the insertion operator with the object of ofstream class (the same way you have been using cout object of iostream class along with << operator) and the data from a file can be read using the extraction operator with the ifstream object(similar to the use of cin object of iostream class and the >> operator

We can also use the get() and put() function to read from and write to a file single character at a time. The following example illustrates the use of the extraction(>>), insertion(>>) operator and get(), put(), getline () functions.

Operator	Calling the put() and get()functions
Insertion operator (<<)	Char name[]=”Ravinder”; ofstream obj; obj>>name; // will write Ravinder into the file

Extraction operator (>>)	char name [size]
	ifstream obj;
	obj>>name;
	cout <<name;

Ex: Use of the insertion operator

```
#include<fstream.h>
void main()
{
        ofstream obj;
        obj.open("ABC.txt");
        obj<<"Hello";           // writes the string “Hello” into the file
        obj.close();
}
```

In the above program, the method open of the fstream class is invoked by the ofstream to open a file. The second line uses the object and the insertion operator as (obj<<”Hello”) to read the stream (called the output stream) from the c++ program and writes to the file named “ABC.txt”.

Our next example illustrates how to read

Ex: Use of extraction operator

```
#include<fstream.h>
void main()
{
        char name[10];
        ifstream obj;
        obj.open("ABC.txt");
        obj>>name;
        cout <<name;
        obj.close();
}
```

File Open Modes

Ex:- WAP in C++ to write roll number and name of 5 students into a file named std_data.txt

```
#include<fstream.h>
#include<iostream.h>
#include<conio.h>
void main()
{
     int roll, i;
     char name[20];
```

```
       ofstream obj;
       obj.open("textFile.txt", ios::out);
       clrscr();
 for (i=0; i<4;i++)
 {
            cout<<"\nenter the name";
            cin>>name;
           cout<<"\n enter roll";
            cin>>roll;
            obj<<"\n"<<roll<<"\t ";
            obj<<name;
 }
            obj.close();
}
```

Ex:- WAP in C++ to add the data of 3 students at the end of a existing file std_data.txt (opened in above example)

Hint: Change the file mode in the open method from ios::out to ios::app. This will allow you to enter the data (roll and std_name at the end of the file std_data.txt

Ex: WAP a program to write data to a file read the data from the same file “std_data.txt” and display on the screen.

The put() function is defined in the ostream class. SO the function put() can be called through the object of ostream class or from any of class that are derived from ostream class. So we can call the metod put() in the following two manners:

	Calling the put() and get()functions	

Put()	//used to write the character value to a file
	ofstream obj;
	obj.put(‘a’); // will write ‘a’ into the file
		for(int i=0; i<20;i++)
		 {
		     obj1.put(data[i]);
		  }
		cout.put(‘a’);

Get()	//used to read the character value from file
	ifstream obj;
	obj.get(ch);
		cin.get(ch); // returns the read character in ch

Read()	//Used to read numeric data from a file	syn
Write()	//Used to write numeric data to a file	

In the following program, we make use of the get(ch) function. The get() function is defined in the istream class. It reads only one byte from the input file and stores in the char variable that is passed to it as argument.

```
#include<fstream.h>
#include<iostream.h>
#include<conio.h>
void main()
{
      char c;
       ifstream obj1;
       obj1.open("textFile.txt"ios::in);
       clrscr();
 do
 {
             obj1.get(c);
            cout<<c;
 }while(obj1);
     obj1.close();
}
```

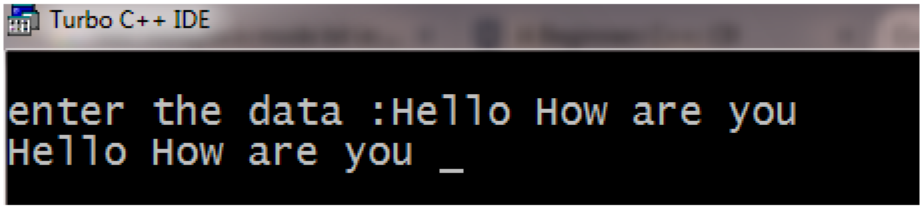
Open Mode Bits:

The open method may also be passed with a second argument called the mode bits. It is an integer type value. There are a number of integer type constant defined in stream handling library. Each of these constant, when passed to open () function as 2nd argument produces different effect while opening the file. These mode bits are given below:

Constant (Mode Bits)	Meaning
ios::app	To append at the end of a given file
ios::ate	To go to the end of a file while opening
ios::binary	To open a binary file
ios::in	To open a file for reading
ios::nocreate	Open () fail if file does not exist
ios::noreplace	Open () fail if file already exist
ios::out	Open a file in write mode
ios::trun	Deletes the content of file if exist

Write a program to enter a string and write it to the file, then open the file to read the string from it and display it on the screen. Use the getline(array, size) function to enter the string from keyboard and get() function to read from the file.

```
#include<fstream.h>
#include<iostream.h>
#include<conio.h>
void main()
{
             int roll;
             char data[20];
            ofstream obj1;
             obj1.open("textFile.txt", ios :: out);
             clrscr();
             cout<<"\nenter the data :";
             cin.getline(data,20);
             obj1<<data;
            obj1.close();
// Open the file for reading
            ifstream obj2;
            obj2.open("textFile.txt", ios :: in);
            char ch;
            do
                        {           obj2.get(ch);                  // or ch = obj2.get()
                                    cout<<ch;
                        } while(!obj2.eof());
            obj2.close();
getch();
}
```



```
enter the data :Hello How are you
Hello How are you _
```

WAP in c++ to read a string from an existing file using the getline() function.

```
#include<conio.h>
void main()
{
            char name[20];
            ifstream obj2;
            obj2.open("textFile.txt", ios::in);
            obj2.getline(name, 20);
            cout<<name;
            obj2.close();
            getch();
}
```

References:

1. https://scs.senecac.on.ca/~chris.szalwinski/archives/btp300.072/content/files.html
2. Objected oriented programming with C++, by Sourav Sahay, Oxford University press, 16th impression 2010
3. Objected oriented programming with C++, by Balagurswamy
4. http://www.cplusplus.com/doc/tutorial/files/