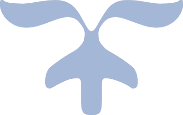


LAB 15

**File I/O**



**NIDA MUNAWAR**

# Introduction to filing

Files are used to store data in a storage device permanently.

File handling provides a mechanism to store the output of a program in a file and to perform various operations on it.

# Streams

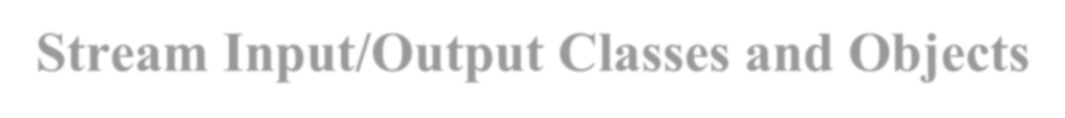
C++ I/O occurs in **streams**, which are sequences of bytes.

In **input operations**, the bytes flow from a device (e.g., a keyboard, a disk drive, a

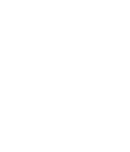
network connection, etc.) to main memory.

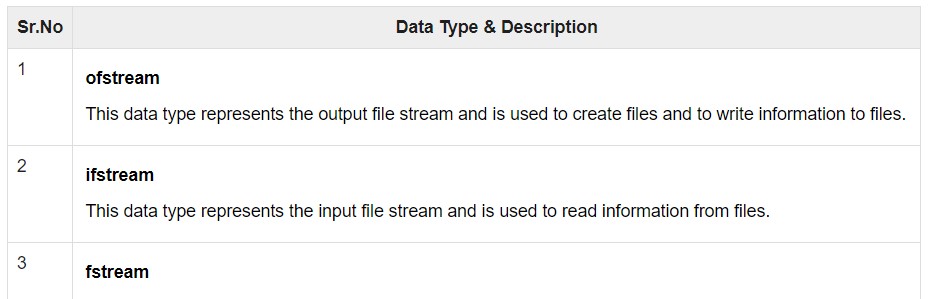
In **output operations**, bytes flow from main memory to a device (e.g., a display screen, a printer, a disk drive, a network connection, etc.).

So far, we have been using the **iostream** standard library, which provides **cin** and **cout** methods for reading from standard input and writing to standard output respectively.

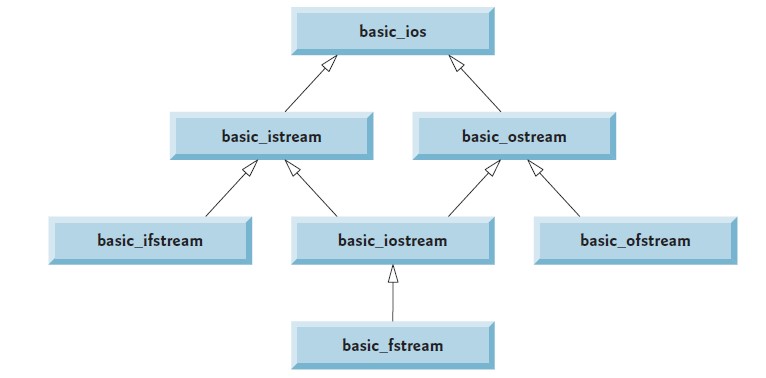


**Stream Input/Output Classes and Objects**





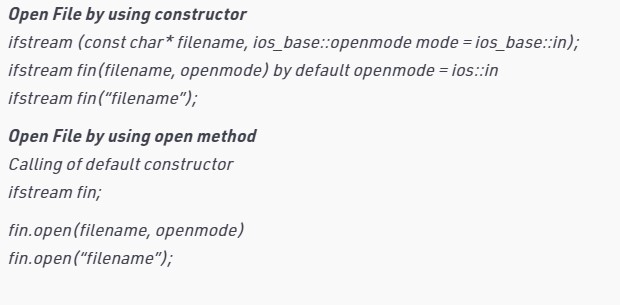
# Stream-I/O template hierarchy



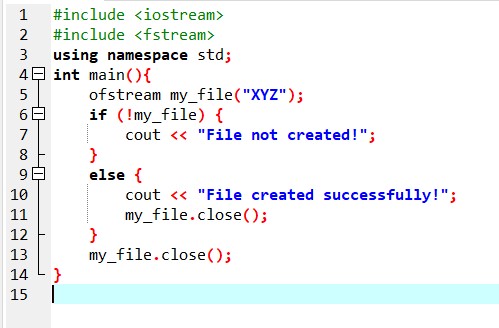
***Opening a File***

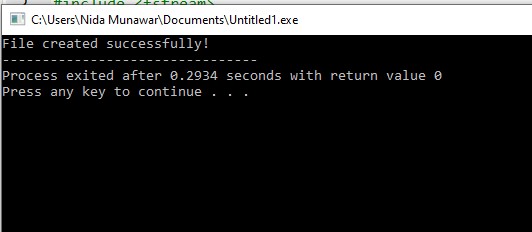
Now the first step to open the particular file for read or write operation. We can open file by

1. passing file name in constructor at the time of object creation
2. using the open method



**File open using constructor method**





**File open using open() method**

The three objects, that is, fstream, ofstream, and ifstream, have the open() function defined in them. The function takes this syntax:

## open (file\_name, mode);

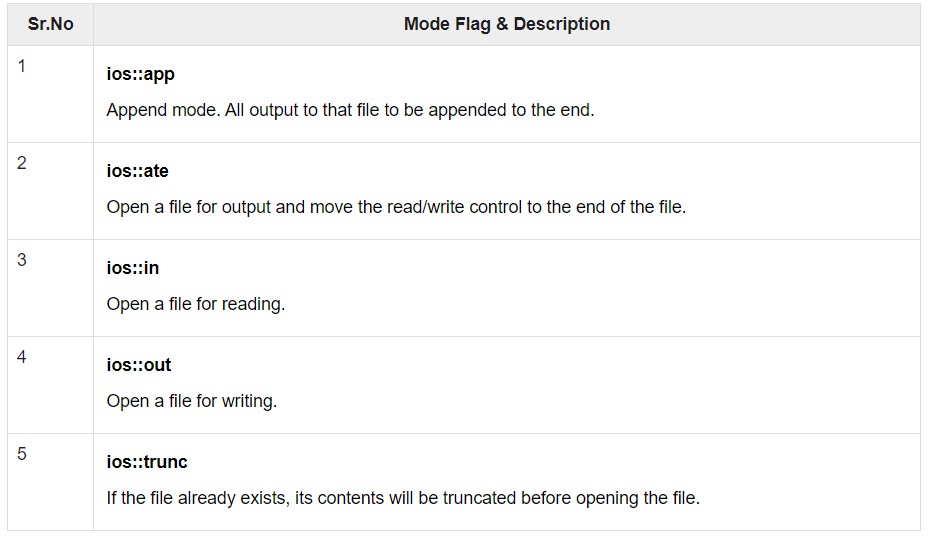
|  |
| --- |
|  |

The file\_name parameter denotes the name of the file to open.

The mode parameter is optional. It can take any of the following values:



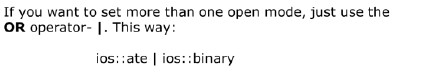
## Mode Flag & Description

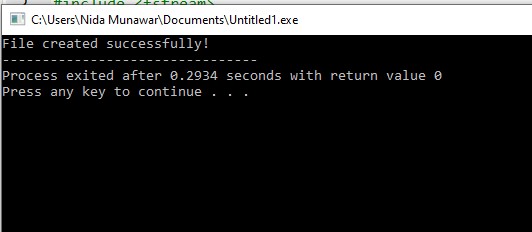
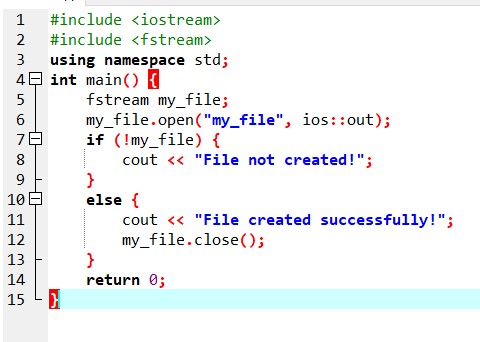


[**Difference between ios::app and ios::ate**](https://stackoverflow.com/questions/10359702/c-filehandling-difference-between-iosapp-and-iosate)

When ios::ate (also called update) is set, the initial position will be the end of the file, but you are free to seek thereafter. When ios::app is set, all output operations are performed at the end of the file. Since all writes are implicitly preceded by seeks, there is no way to write elsewhere.

### File Open Mode

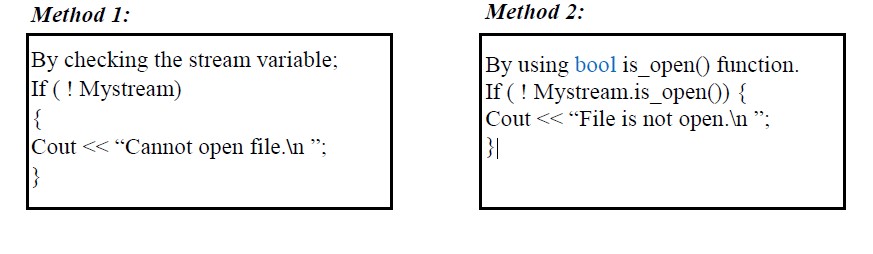




### Open()

* Opening a file associates a file stream variable declared in the program with a physical file at the source, such as a disk.
* In the case of an input file:
* the file must exist before the open statement executes.
* If the file does not exist, the open statement fails and the input stream enters the fail state
* An output file does not have to exist before it is opened;
* if the output file does not exist, the computer prepares an empty file for output.
* If the designated output file already exists, by default, the old contents are erased when the file is opened.

### Validate the file before trying to access



***File Processing Function***

* open(): To create a file
* close(): To close an existing file
* get(): to read a single character from the file
* put(): to write a single character in the file



* read(): to read data from a file..>>
* write(): to write data into a file.. <<

#### How to Close Files

Once a C++ program terminates, it automatically

* flushes the streams
* releases the allocated memory
* closes opened files.

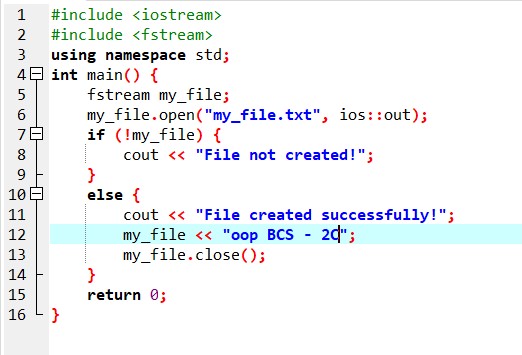
However, as a programmer, you should learn to close open files before the program terminates.

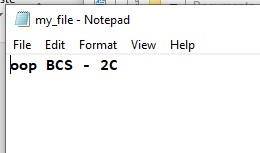
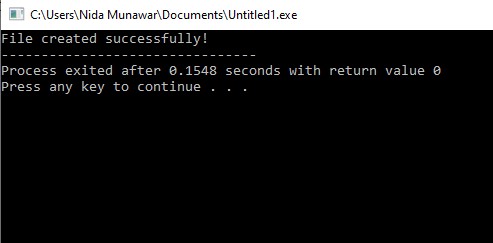
The fstream, ofstream, and ifstream objects have the close() function for closing files. The function takes this syntax:

**void close();**

#### How to Write to Files

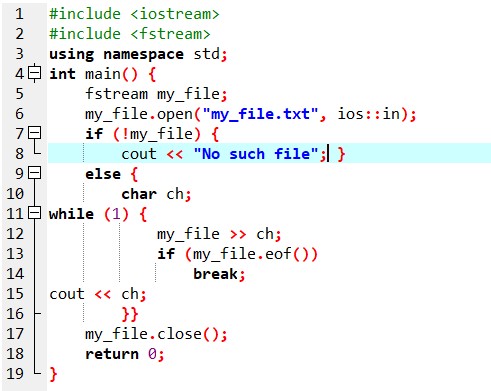
You can write to file right from your C++ program. You use stream insertion operator (<<) for this. The text to be written to the file should be enclosed within double-quotes.





#### How to Read from Files

You can read information from files into your C++ program. This is possible using stream extraction operator (>>). You use the operator in the same way you use it to read user input from the keyboard. However, instead of using the cin object, you use the ifstream/ fstream object.



1. an else statement to state what to do if the file is found.
2. Create a char variable named ch.
3. Create a while loop for iterating over the file contents.
4. Write/store contents of the file in the variable ch.
5. Use an if condition and eof() function that is, end of the file, to ensure the compiler keeps on reading from the file if the end is not reached.
6. Use a break statement to stop reading from the file once the end is reached.
7. Print the contents of variable ch on the console.
8. End of the while body.

### Using Member Function getline

as it name states, read a whole line, or at least till a delimiter that can be specified.

**Syntax:**

istream& getline(istream& is, string& str, char delim);

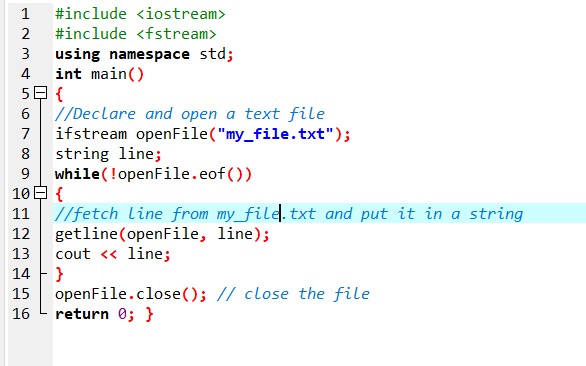
**Parameters:**

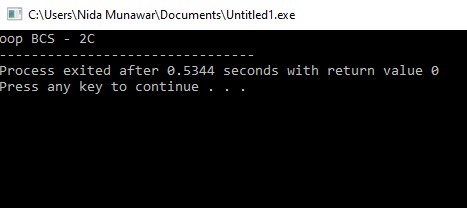
* **is:** It is an object of istream class and tells the function about the stream from where to read the input from.
* **str:** It is a string object, the input is stored in this object after being read from the stream.
* **delim:** It is the delimitation character which tells the function to stop reading further input after reaching this character.

**Syntax:** istream& getline (istream& is, string& str);

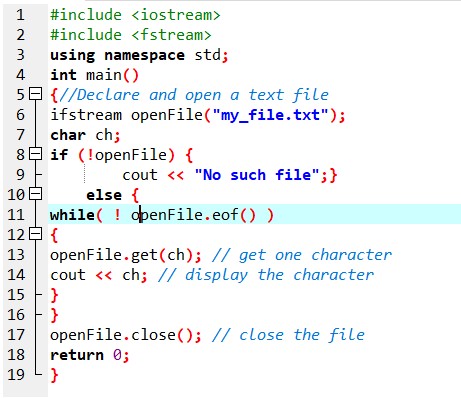
The second declaration is almost the same as that of the first one. The only difference is, the latter have a delimitation character which is by default new line(\n) character.

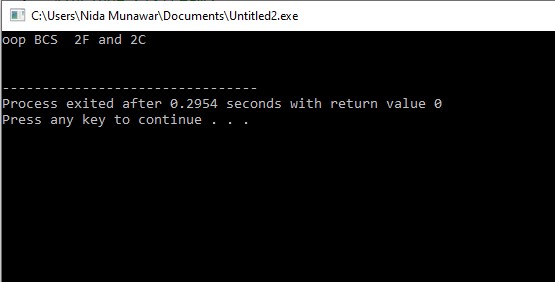
### Read a line



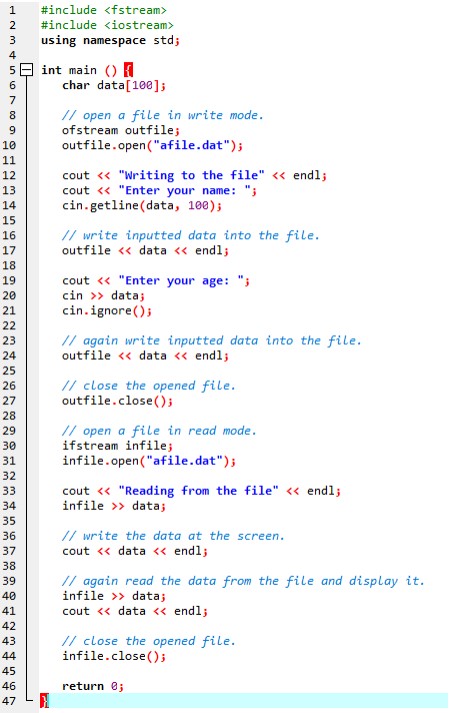


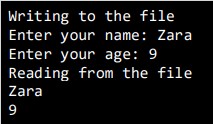
### Read character by character





### Example





Examples make use of additional functions from cin object, like getline() function to read the line from outside and ignore() function to ignore the extra characters left by previous read statement

### write() function

The write() function is used to write object or record (sequence of bytes) to the file. A record may be an array, structure or class. Syntax of write() function

fstream fout;

fout.write( (char \*) &obj, sizeof(obj) );

The write() function takes two arguments. **&obj :** Initial byte of an object stored in memory. **sizeof(obj) :** size of object represents the total number of bytes to be written from initial byte.

## *Using the read() and write() function for binary I/O.*

//Writing a class object to a file using ofstream class and mode ios::out

#include<iostream>

#include<fstream>

using namespace std;

const int size = 3;

class A

{

private:

char name[40];

int age;

float height;

char gender;

char newline\_chr;

public:

void putdata();

void getdata();

};

//Defining the function putdata() to allow user to enter the data member of a object.

void A :: putdata()

{

cout<<"Enter the name : ";

cin.getline(name,40);

cout<<"Enter the age : ";

cin>>age;

cout<<"Enter the height : ";

cin>>height;

cout<<"Enter the gender : ";

cin>>gender;

//This one captures the enter key(newline character) pressed after entering the gender.

cin.get(newline\_chr);

}

void A :: getdata()

{

cout<<"The name is : " << name << "\n";

cout<<"The age is : " << age << "\n";

cout<<"The height is : " << height << "\n";

cout<<"The gender is : " << gender << "\n";

}

int main()

{

//Creating an output stream

ofstream fstream\_ob;

//Calling the open function to read and write an object to/from a file

fstream\_ob.open("File1.txt", ios::out| ios:: app);

//Creating an array of objects of class A

A ob1[size];

//Calling the putdata() function

for(int i=0;i<size;i++)

{

//Calling putdata() to let user enter the values for data member of an object.

ob1[i].putdata();

}

//Calling the write() function to write an array of objects to a file in a binary form.

fstream\_ob.write( (char \*) & ob1, sizeof(ob1));

cout<<"Congrats! Your array of objects is successfully written to the file \n";

//Closing the output stream

fstream\_ob.close();

//Creating an intput stream

ifstream ifstream\_ob;

//Calling the open function to read and write an object to/from a file

ifstream\_ob.open("File1.txt", ios::in);

cout<<"\nReading an array of objects from a file : \n";

//Calling the read() function to read an array of objects from a file and transfer its content to an empty object

ifstream\_ob.read( (char \*) & ob1, sizeof(ob1));

for(int i=0;i<size;i++)

{

//Calling getdata() to read the values of an object just read

ob1[i].getdata();

}

//Closing the input stream

ifstream\_ob.close();

return 0;

}

**Another Example of write() function**

#include <iostream> #include <fstream> using namespace std; class student{ int roll; char name[25]; float marks; public:

void getdata(){ cout << "enter roll no" << endl;

cin >> roll;

cout << "enter name" << endl;

cin >> name;

cout << "enter marks" << endl; cin >> marks;

}

void addRecord()

{

fstream f; student s;

f.open("studen.dat",ios::app | ios::binary );

s.getdata();

f.write((char\*)&s, sizeof(s));

f.close();

}

};

int main(){

student s;

char c = 'n';

do{

s.addRecord();

cout << "do you want to add another record";

cin >> c;

} while (c == 'y' || c == 'Y'); cout << "data written successfully";

}

### read() function

The read() function is used to read object (sequence of bytes) to the file. Syntax of read() function

fstream fin; fin.read( (char \*) &obj, sizeof(obj) );

The read() function takes two arguments. **&obj :** Initial byte of an object stored in file. **sizeof(obj) :** size of object represents the total number of bytes to be read from initial byte.

The read() function returns NULL if no data read.

Example of read() function

#include <iostream> #include <fstream> using namespace std; class student{

int roll; char name[25]; float marks; public:

void displayStudent(){ cout<<"Roll no: "<<roll<<endl <<"NAME: "<<name<<endl

<<"Marks: "<<marks<<endl;

}

void Readdata()

{

fstream f; student s;

f.open("studen.dat",ios::in | ios::binary);

if(f.read((char\*)&s,sizeof(s))){

cout<<endl<<endl;

s.displayStudent();

}

else{

cout<<"Error in reading data from file...\n";

}

}

};

int main(){

student s;

s.Readdata();

return 0;

}

**Exercise**

#### QUESTION#1

Write a program to implement I/O operations on characters. I/O operations includes inputting a string, calculating length of the string, Storing the String in a file and fetch the stored characters from it.

**QUESTION#2**

Write a program to copy the contents of one file to another.

#### QUESTION#3

Take a class Person having two attributes name and age.

Include a parametrized constructor to give values to all data members. In main function

1. Create an instance of the person class and name it person1.
2. Create a binary file person.bin and write person1 object into it.
3. Read the person1 object from the file.
4. Return 0

#### QUESTION#4

Take a class Participant having three attributes (ID, name and score) and following member functions

 Input () function takes data of the object and stores it in a file name participant.dat

 Output () function takes id from user and show respective data of that id.

 Max () gives the highest score of the Participant in the file.

**QUESTION#6**

Write a function in C++ to count and display the number of lines not starting with alphabet 'A' present in a text file "STORY.TXT". Example:

1. If the file "STORY.TXT" contains the following lines,
2. The rose is red.
3. A girl is playing there.
4. There is a playground.
5. An aeroplane is in the sky.
6. Numbers are not allowed in the password.
7. The function should display the output as 3.