Name:- PRANAY JHA
Reg no.:- RA2111003011092
Course:- OOPS
Section:- B2

DISK FILE HANDLING READING AND WRITING DATA

File handling is used to store data permanently in a computer. Using file handling we can store our data in secondary memory (Hard disk).
How to achieve the File Handling
For achieving file handling we need to follow the following steps:-
STEP 1-Naming a file
STEP 2-Opening a file
STEP 3-Writing data into the file
STEP 4-Reading data from the file
STEP 5-Closing a file.

Streams in C++ :-

We give input to the executing program and the execution program gives back the output. The sequence of bytes given as input to the executing program and the sequence of bytes that comes as output from the executing program are called stream. In other words, streams are nothing but the flow of data in a sequence.

The input and output operation between the executing program and the devices like keyboard and monitor are known as “console I/O operation”. The input and output operation between the executing program and files are known as “disk I/O operation”.

Classes for File stream operations :-

The I/O system of C++ contains a set of classes which define the file handling methods. These include ifstream,

ofstream and fstream classes. These classes are derived from fstream and from the corresponding iostream class. These classes, designed to manage the disk files, are declared in fstream and therefore we must include this file in any program that uses files.

1. ios:-
    - ios stands for input output stream.
    - This class is the base class for other classes in this class hierarchy.
    - This class contains the necessary facilities that are used by all the other derived classes for input and output operations.
2. istream:-
    - istream stands for input stream.
    - This class is derived from the class ‘ios’.
    - This class handle input stream.
    - The extraction operator(>>) is overloaded in this class to handle input streams from files to the program execution.
    - This class declares input functions such as get(), getline() and read().
3. ostream:-
    - ostream stands for output stream.
    - This class is derived from the class ‘ios’.
    - This class handle output stream.
    - The insertion operator(<<) is overloaded in this class to handle output streams to files from the program execution.
    - This class declares output functions such as put() and write().
4. streambuf:-
    - This class contains a pointer which points to the buffer which is used to manage the input and output streams.
5. fstreambase:-
    - This class provides operations common to the file streams. Serves as a base for fstream, ifstream and ofstream class.
    - This class contains open() and close() function.
6. ifstream:-

- This class provides input operations.
- It contains open() function with default input mode.
- Inherits the functions get(), getline(), read(), seekg() and tellg() functions from the istream.

7. ofstream:-

- This class provides output operations.
- It contains open() function with default output mode.
- Inherits the functions put(), write(), seekp() and tellp() functions from the ostream.

8. fstream:-

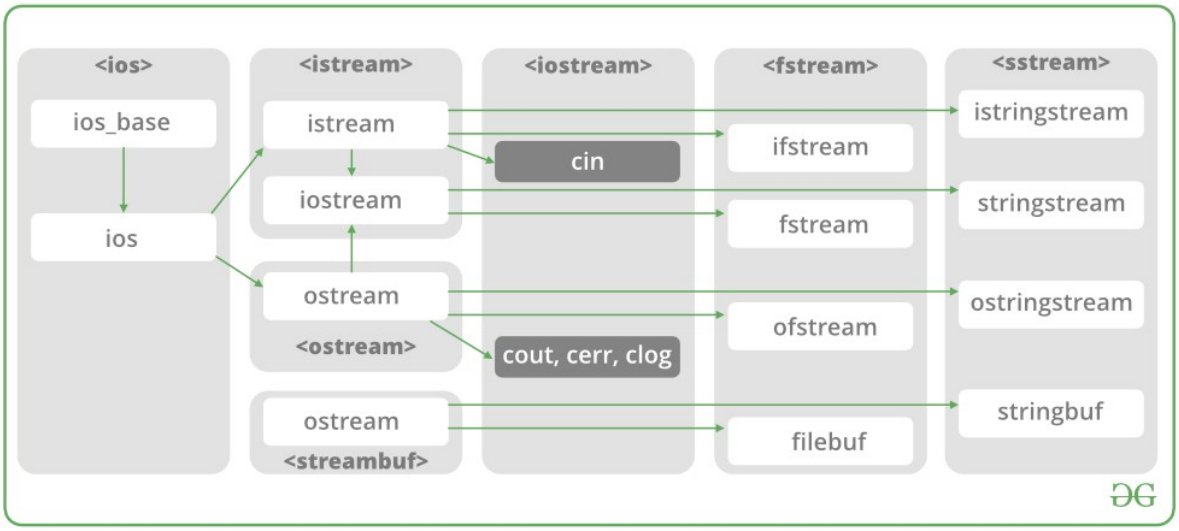
- This class provides support for simultaneous input and output operations.
- Inherits all the functions from istream and ostream classes through iostream.

9. filebuf:-

- Its purpose is to set the file buffers to read and write.
- We can also use file buffer member function to determine the length of the file.

In C++, files are mainly dealt by using three classes fstream, ifstream, ofstream available in fstream headerfile.

ofstream: Stream class to write on files **ifstream:** Stream class to read from files **fstream:** Stream class to both read and write from/to files.



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

For e.g.

Open File by using constructor

ifstream (const char filename, ios_base::openmode mode = ios_base::in);*

ifstream fin(filename, openmode) by default openmode = ios::in

ifstream fin("filename"); ***Open File by using open method***

Calling of default constructor

ifstream fin;

fin.open(filename, openmode)

fin.open("filename");

Modes :

Member Constant	Stands For	Access
in *	input	File open for reading: the internal stream buffer supports input operations.
out	output	File open for writing: the internal stream buffer supports output operations.
binary	binary	Operations are performed in binary mode rather than text.
ate	at end	The output position starts at the end of the file.
app	append	All output operations happen at the end of the file, appending to its existing contents.
trunc	truncate	Any contents that existed in the file before it is open are discarded.

Default Open Modes :

ifstream ios::in

ofstream ios::out

fstream ios::in | ios::out

Below is the implementation by using **ifstream & ofstream classes.**

. C++

```
/* File Handling with C++ using ifstream & ofstream class object*/
/* To write the Content in File*/
/* Then to read the content of file*/
#include <iostream>

/* fstream header file for ifstream, ofstream,
fstream classes */
#include <fstream>
  using namespace
std;
// Driver Code int
main()
{
    // Creation of ofstream class object
ofstream fout;
      string
line;
    // by default ios::out mode, automatically deletes
    // the content of file. To append the content, open in ios:app
    // fout.open("sample.txt", ios::app)
fout.open("sample.txt");
    // Execute a loop If file successfully opened
while (fout) {
        // Read a Line from standard input
getline(cin, line);
        // Press -1 to exit
if (line == "-1")
break;
        // Write line in file
fout << line << endl;
    }

    // Close the File
fout.close();
    // Creation of ifstream class object to read the file
ifstream fin;
    // by default open mode = ios::in mode
fin.open("sample.txt");
    // Execute a loop until EOF (End of File)
while (getline(fin, line)) {
```

```
        // Print line (read from file) in Console
cout << line << endl;
    }

    // Close the file
fin.close();

    return 0;
}
```

Below is the implementation by using **fstream class.**

. C++

```
/* File Handling with C++ using fstream class object */
/* To write the Content in File */
/* Then to read the content of file*/
#include <iostream>

/* fstream header file for ifstream, ofstream,
fstream classes */
#include <fstream>

using namespace std;
// Driver Code int
main()
{
    // Creation of fstream class object
fstream fio;

    string line;

    // by default openmode = ios::in|ios::out mode
    // Automatically overwrites the content of file, To append
    // the content, open in ios:app
    // fio.open("sample.txt", ios::in|ios::out|ios::app)
// ios::trunc mode delete all content before open
fio.open("sample.txt", ios::trunc | ios::out | ios::in);
    // Execute a loop If file successfully Opened
while (fio) {
        // Read a Line from standard input
getline(cin, line);
        // Press -1 to exit
if (line == "-1")
break;
        // Write line in file
fio << line << endl;
    }
```

```
    // Execute a loop until EOF (End of File)
// point read pointer at beginning of file
fio.seekg(0, ios::beg);
      while
(fio) {
        // Read a Line from File
getline(fio, line);
        // Print line in Console
cout << line << endl;
    }

    // Close the file
fio.close();

    return 0;
}
```

- C++

Q: write a single file handling program in c++ to reading and writing data on a file.

```
#include<iostream>
#include<fstream>
    using namespace
std; main() {
      int rno,fee;
char name[50];

      cout<<"Enter the Roll Number:";
cin>>rno;
      cout<<"\nEnter the Name:";
cin>>name;
      cout<<"\nEnter the Fee:";
cin>>fee;
          ofstream
fout("d:/student.doc");
          fout<<rno<<"\t"<<name<<"\t"<<fee;   //write data to the file
student

fout.close();
          ifstream
fin("d:/student.doc");
          fin>>rno>>name>>fee;   //read data from the file
student

fin.close();

cout<<endl<<rno<<"\t"<<name<<"\t"<<fee;

    return 0;
}
```

```
. C++
// Q: WA C++ file handling program to read data from the file called
student.doc
#include<iostream>
#include<fstream>
  using namespace
std;
  main()
{
      int rno,fee;
char name[50];
          ifstream
fin("d:/student.doc");
          fin>>rno>>name>>fee;   //read data from the file
student

fin.close();

cout<<endl<<rno<<"\t"<<name<<"\t"<<fee;

return 0;
}
```

Given a file “Input.txt” in which every line has values same as instance variables of a class.
Read the values into the class’s object and do necessary operations.

Theory :

The data transfer is usually done using '>>'

and <<' operators. But if you have a class

with 4 data members and want to write all 4

data members from its object directly to a

file or vice-versa, we can do that using

following syntax :

To write object's data members in a file :

```
// Here file_obj is an object of ofstream
file_obj.write((char *) & class_obj, sizeof(class_obj));
```

To read file's data members into an object :

```
// Here file_obj is an object of ifstream
file_obj.read((char *) & class_obj, sizeof(class_obj));
```

Examples:

Input :

Input.txt :

- C++

```
Michael 19 1806
Kemp 24 2114
Terry 21 2400
Operation : Print the name of the highest
rated programmer.
```

Output :

```
Terry
```

```
// C++ program to demonstrate read/write of class
// objects in C++.
#include  <iostream>
#include   <fstream>
using namespace std;
// Class to define the properties
class Contestant { public:
    // Instance variables
string Name;     int Age,
Ratings;
    // Function declaration of input() to input info
int input();
    // Function declaration of output_highest_rated() to
    // extract info from file Data Base
int output_highest_rated();
};

// Function definition of input() to input info
int Contestant::input()
{
    // Object to write in file
ofstream file_obj;
    // Opening file in append mode
file_obj.open("Input.txt", ios::app);
    // Object of class contestant to input data in file
Contestant obj;

    // Feeding appropriate data in variables
string str = "Michael";     int age = 18,
ratings = 2500;
    // Assigning data into object
obj.Name = str;     obj.Age =
age;     obj.Ratings = ratings;

    // Writing the object's data in file
file_obj.write((char*)&obj, sizeof(obj));
```

```
    // Feeding appropriate data in variables
str = "Terry";      age = 21;      ratings =
3200;
    // Assigning data into object
obj.Name = str;      obj.Age =
age;      obj.Ratings = ratings;
    // Writing the object's data in file
file_obj.write((char*)&obj, sizeof(obj));

    return 0;
}

// Function definition of output_highest_rated() to
// extract info from file Data Base int
Contestant::output_highest_rated()
{
    // Object to read from file
ifstream file_obj;
    // Opening file in input mode
file_obj.open("Input.txt", ios::in);
    // Object of class contestant to input data in file
Contestant obj;

    // Reading from file into object "obj"
file_obj.read((char*)&obj, sizeof(obj));
    // max to store maximum ratings
int max = 0;
    // Highest_rated stores the name of highest rated contestant
string Highest_rated;
    // Checking till we have the feed
while (!file_obj.eof()) {          //
Assigning max ratings              if
(obj.Ratings > max) {
            max = obj.Ratings;
            Highest_rated = obj.Name;
        }

        // Checking further
file_obj.read((char*)&obj, sizeof(obj));
    }

    // Output is the highest rated contestant
cout << Highest_rated;      return 0;
}

// Driver code
int main()
{
    // Creating object of the class
Contestant object;
```

```
    // Inputting the data
object.input();
    // Extracting the max rated contestant
object.output_highest_rated();
      return
0;
}
```

Output: Terry

THANK YOU