File Handling through C++ Classes

Difficulty Level : Medium • Last Updated : 02 Nov, 2022

Read Discuss Courses Practice Video

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 :-

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 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.

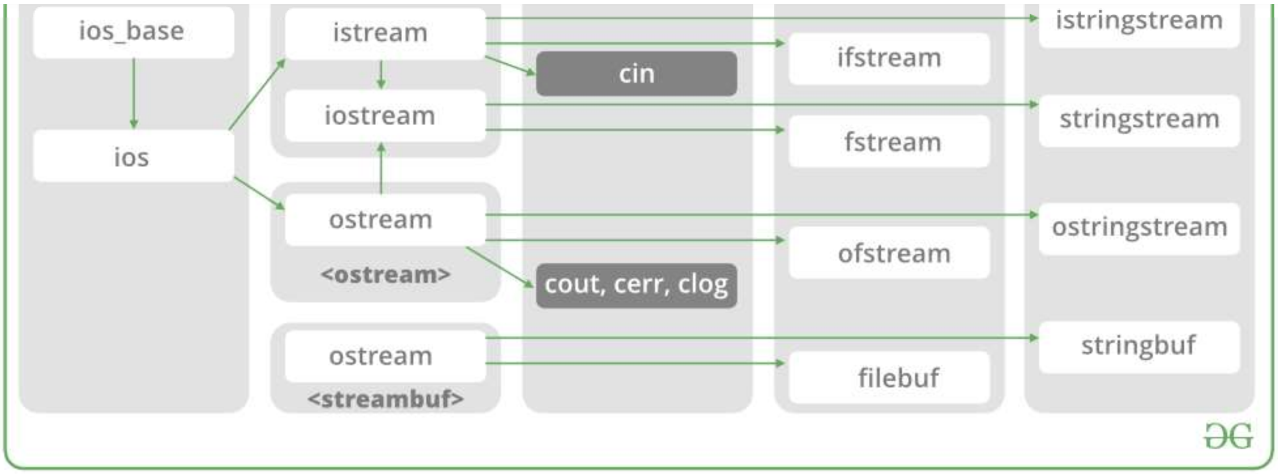
8. fstream:-

- 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 :

Constant	For	
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

Problem Statement : To read and write a File in C++.

Examples:

```
-1
Output :
Welcome in GeeksforGeeks. Best way to learn things.
```

Recommended: Please try your approach on ***{IDE}*** first, before moving on to the solution.

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);
        // 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()
```

```
    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);
```

DSA Data Structures Algorithms Interview Preparation Data Science Topic-wise Practice

```
        // Read a Line from File
        getline(fio, line);

        // Print line in Console
        cout << line << endl;
    }

    // Close the file
    fio.close();

    return 0;
}
```

```
#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;
```

```
    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;
}
```

Related Articles

1. Exception Handling using classes in C++
2. Four File Handling Hacks which every C/C++ Programmer should know
3. Set Position with seekg() in C++ File Handling
4. tellp() in file handling with c++ with example
5. How to work with file handling in C++
6. Error handling during file operations in C/C++
7. Contact Book in C++ using File Handling
8. ATM using file handling in C++
9. Bookshop management system using file handling

Like 192

Previous

Next

Article Contributed By :

prakhargvp
@prakhargvp

Vote for difficulty

Current difficulty : Medium

Easy Normal Medium Hard Expert

Improved By : ruhelaa48, anikaseth98, anuragmourya975, aditiyadav20102001, it2024002, ayushknath, jagadeeshprasad

Article Tags : cpp-file-handling, C++

Practice Tags : CPP

Improve Article

Report Issue

Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

About Us

Careers

In Media

Contact Us

Privacy Policy

Copyright Policy

Advertise with us

Learn

DSA

Algorithms

Data Structures

SDE Cheat Sheet

Machine learning

CS Subjects

Video Tutorials

Courses

News

Top News

Technology

Work & Career

Business

Finance

Lifestyle

Knowledge

Languages

Python

Java

CPP

Golang

C#

SQL

Kotlin

Web Development

Web Tutorials

Django Tutorial

HTML

JavaScript

Bootstrap

Contribute

Write an Article

Improve an Article

Pick Topics to Write

Write Interview Experience

Internships