Managing Console I/O operations in C++

Every program takes some data as input and generates processed data as an output following the familiar input process output cycle. It is essential to know how to provide the input data and present the results in the desired form. The use of the **cin** and **cout** is already known with the operator **>>** and **<<** for the input and output operations. In this article, we will discuss how to control the way the output is printed.

C++ supports a rich set of I/O functions and operations. These functions use the advanced features of C++ such as classes, derived classes, and virtual functions. It also supports all of C's set of I/O functions and therefore can be used in C++ programs, but their use should be restrained due to two reasons.

1. I/O methods in C++ support the concept of OOPs.
2. I/O methods in C cannot handle the user-define data type such as class and object.

It uses the concept of stream and stream classes to implement its I/O operations with the console and disk files.

C++ Stream

The I/O system in C++ is designed to work with a wide variety of devices including terminals, disks, and tape drives. Although each device is very different, the I/O system supplies an interface to the programmer that is independent of the actual device being accessed. This interface is known as the **stream.**

- A stream is a sequence of bytes.
- The source stream that provides the data to the program is called the input stream.
- The destination stream that receives output from the program is called the output stream.
- The data in the input stream can come from the keyboard or any other input device.
- The data in the output stream can go to the screen or any other output device.

C++ contains several pre-defined streams that are automatically opened when a program begins its execution. These include **cin** and **cout**. It is known that **cin** represents the input stream connected to the standard input device (usually the keyboard) and **cout** represents the output stream connected to the standard output device (usually the screen).

C++ Stream Classes

The C++ I/O system contains a hierarchy of classes that are used to define various streams to deal with both the console and disk files. These classes are called stream classes. The hierarchy of stream classes used for input and output operations is with the console unit. These classes are declared in the **header file iostream**. This file should be included in all the programs that communicate with the console unit.

```
#include <iostream>
using namespace std;

int main()
{

    cout << " This is my first"
            " Blog on the gfg";
    return 0;
}
```

Output:

```
This is my first Blog on the gfg
```

The **ios** are the base class for **istream** (input stream) and **ostream** (output stream) which are in turn base classes for **iostream** (input/output stream). The class **ios** are declared as the virtual base class so that only one copy of its members is inherited by the iostream.

The class **ios** provide the basics support for formatted and unformatted I/O operations. The class istream provides the facilities formatted and unformatted input while the class ostream (through inheritance) provides the facilities for formatted output.

The class iostream provides the facilities for handling both input and output streams. Three classes add an assignment to these classes:

- **istream_withassign**
- **ostream_withassign**
- **iostream_withassign**

Tabular Representation of stream classes for Console Operation:

Class name	Content
ios (General input/output stream class)	• Contains basic facilities that are used by all other input and output classes • Also contains a pointer to a buffer object (**streambuf** object) • Declares Constants and functions that are necessary for handling formatted input and output operations
istream (Input stream)	• It inherits the properties of **ios** • It declares input functions such as **get(), getline(), and read()** • It contains overload extraction operator **>>**
ostream (Output Stream)	• It inherits the properties of **ios**. • It declares input functions such as **put()** and **write()**. • It contains an overload extraction operator **<<**.
iostream (Input/output stream)	• Inherits the properties of **ios stream** and **ostream** through multiple inheritances and thus contains all the input and output functions.
Streambuf	• It provides an interface to physical devices through buffers. • It acts as a base for filebuf class used ios file.