Header files available in C++ for Input/Output operations are:

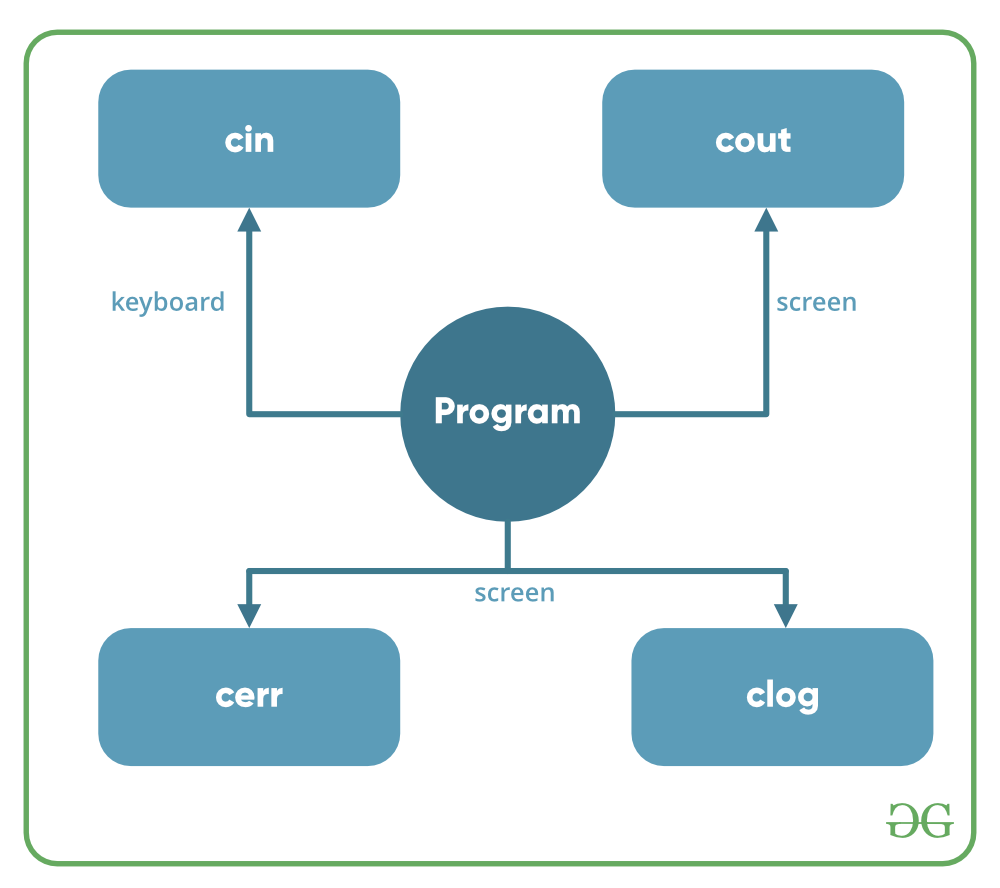
1. **iostream**: iostream stands for standard input-output stream. This header file contains definitions to objects like cin, cout, cerr etc.
2. **iomanip**: iomanip stands for input output manipulators. The methods declared in this files are used for manipulating streams. This file contains definitions of setw, setprecision etc.
3. **fstream**: This header file mainly describes the file stream. This header file is used to handle the data being read from a file as input or data being written into the file as output.

The two keywords **cout in C++** and **cin in C++** are used very often for printing outputs and taking inputs respectively. These two are the most basic methods of taking input and printing output in C++. To use cin and cout in C++ one must include the header file *iostream* in the program.

Basic Input / Output in C++

C++ comes with libraries which provides us with many ways for performing input and output. In C++ input and output is performed in the form of a sequence of bytes or more commonly known as **streams**.

* **Input Stream:** If the direction of flow of bytes is from the device(for example, Keyboard) to the main memory then this process is called input.
* **Output Stream:** If the direction of flow of bytes is opposite, i.e. from main memory to device( display screen ) then this process is called output.



# What is Source Code and Object Code?

## 1: Source Code:

First of all we can discuss about source code. The codes which are written in high level languages are called source code. The high levels languages like Php, C++, Python and Java etc. The source code is easier to understand because this type of codes are near to human language. Instructions in these languages are written in English like words. The computer cannot understand source code. It converts the source code into object code and then execute it.

## 2: Object Code:

The code which are in machine language is called object code. It is also called object program or machine code. Computer easily execute object code because this type of language is written in binary form and the binary language is the fundamental language of computer system.

I hope after reading this article you can know about what is source code and object code. This is my opinion about this topic. If I can take any mistake please comment me and I am waiting for your response.

# debugging

Debugging, in computer programming and engineering, is a multistep process that involves identifying a problem, isolating the source of the problem, and then either correcting the problem or determining a way to work around it. The final step of debugging is to test the correction or workaround and make sure it works.

In [software development](https://whatis.techtarget.com/definition/software-development), the debugging process begins when a developer locates a [code](https://whatis.techtarget.com/definition/code) error in a computer program and is able to reproduce it. Debugging is part of the [software testing](https://whatis.techtarget.com/definition/software-testing) process and is an integral part of the entire software development lifecycle.

In hardware development, the debugging process typically looks for hardware components that are not installed or configured correctly. For example, an engineer might run a JTAG connection test to debug connections on an integrated circuit.

### How debugging works in software

Typically, the debugging process starts as soon as code is written and continues in successive stages as code is combined with other units of programming to form a software product. In a large program that has thousands and thousands of lines of code, the debugging process can be made easier by using strategies such as [unit tests](https://searchsoftwarequality.techtarget.com/definition/unit-testing), code reviews and [pair program](https://searchsoftwarequality.techtarget.com/definition/Pair-programming)

### examples

Some examples of common coding errors include the following:

* Syntax error
* Runtime error
* Semantic error
* Logic error

Types of errors

**Syntax errors**

These are errors where the compiler finds something wrong with your program, and you can't even try to execute it. For example, you may have incorrect punctuation, or may be trying to use a variable that hasn't been declared.

Syntax errors are the easiest to find and correct. The compiler will tell you where it got into trouble, and its best guess as to what you did wrong. Usually the error is on the exact line indicated by the compiler, or the line just before it; however, if the problem is incorrectly nested braces, the actual error may be at the beginning of the nested block.

**Runtime errors**

If there are no syntax errors, Java may detect an error while your program is running. You will get an error message telling you the kind of error, and a **stack trace** that tells not only where the error occurred, but also what other [method](https://www.cis.upenn.edu/~matuszek/General/JavaSyntax/methods.html)  in

**Logic errors**

A logic error, or **bug**, is when your program compiles and runs, but does the wrong thing. The Java system, of course, has no idea what your program is *supposed* to do, so it provides no additional information to help you find the error.

Ways to track down a logic error include:

* Think about what the program must have done in order to produce the results it did. This will lead you to where the error must have occurred.
* Put in [print statements](https://www.cis.upenn.edu/~matuszek/General/JavaSyntax/print-statements.html) to help you figure out what the program is actually doing.
* Use a **debugger** to step through your program and watch what it does.