**C++ Basic Input and Output (I/O)**

In C++, input and output operations are handled through a concept known as *streams*. A stream represents a sequence of bytes flowing between the program and input/output devices such as the keyboard, screen, files, or network connections. C++ provides predefined functions and classes to perform these operations, all of which are included in specific *header files*.

There are two main types of operations:

* **Output operation:** Transfers bytes from the program's memory to an output device (like the display screen).
* **Input operation:** Transfers bytes from an input device (like a keyboard) to the program's memory.

C++ uses a set of predefined header files to manage input and output. Let's explore the most important ones:

**Key Header Files for I/O Operations**

| **Header File** | **Description** |
| --- | --- |
| <iostream> | Defines objects like cout, cin, and cerr for standard input/output streams. |
| <iomanip> | Provides functions for formatted I/O, such as setprecision and setw. |
| <fstream> | Handles file input/output operations using ifstream (input) and ofstream (output). |

**1. <iostream> Header File**

The <iostream> header file is essential for console I/O operations. It defines several classes, including:

* istream for input streams.
* ostream for output streams.

The most commonly used objects from <iostream> are cin (for input) and cout (for output).

**Example: Console Input/Output**

cpp

**#include <iostream>**

**using namespace std;**

**int main() {**

**int num;**

**cout << "Enter a number: "; // Output prompt**

**cin >> num; // Input from the user**

**cout << "You entered: " << num << endl; // Output result**

**return 0;**

**}**

**Output:**

**Enter a number: 42**

**You entered: 42**

**2. <iomanip> Header File**

The <iomanip> header file allows for *formatted input and output*. It provides functions to control the precision, width, and alignment of output. Frequently used functions include:

* setw: Sets the width of the output field.
* setprecision: Sets the number of digits displayed after the decimal point.
* fixed: Ensures floating-point values are displayed in fixed-point notation.

**Example: Formatted Output**

cpp

#include <iostream>

#include <iomanip>

using namespace std;

int main() {

double pi = 3.14159;

cout << fixed << setprecision(2) << "Value of pi: " << pi << endl;

return 0;

}

**Output:**

Value of pi: 3.14

**3. <fstream> Header File**

The <fstream> header file is used for file I/O operations. It provides:

* ifstream for input from files.
* ofstream for output to files.

**Example: File Output**

cpp

#include <iostream>

#include <fstream>

using namespace std;

int main() {

ofstream outputFile("output.txt"); // Open a file for writing

if (outputFile.is\_open()) {

outputFile << "Hello, File I/O!"; // Write to the file

outputFile.close(); // Close the file

cout << "File is written successfully." << endl;

} else {

cout << "Failed to open the file." << endl;

}

return 0;

}

**Output:**

File is written successfully.

**Namespace and using Directive**

In C++, the Standard Library components are part of the std namespace. To access objects like cout, cin, or endl, you need to prefix them with std::. However, by using the directive using namespace std;, you can omit this prefix, making your code cleaner and more readable.

**Without using namespace std:**

cpp

#include <iostream>

int main() {

std::cout << "Hello, world!" << std::endl;

return 0;

}

**With using namespace std:**

cpp

#include <iostream>

using namespace std;

int main() {

cout << "Hello, world!" << endl;

return 0;

}

**Standard Streams in C++**

1. **cout: Standard Output Stream**  
   The cout object is used for output, typically to the console. It uses the insertion operator <<.

cpp

cout << "Hello, C++!" << endl;

1. **cin: Standard Input Stream**  
   The cin object reads input from the user, typically from the keyboard. It uses the extraction operator >>.

cpp

int age;

cin >> age;

1. **endl: New Line and Stream Flush**  
   The endl object adds a newline character and flushes the output stream.

cpp

cout << "First Line" << endl << "Second Line" << endl;

1. **cerr: Standard Error Stream**  
   The cerr object is used for unbuffered error messages. It displays output immediately.

cpp

cerr << "This is an error message." << endl;

1. **clog: Standard Log Stream**  
   The clog object is a buffered output stream, typically used for diagnostic or logging information.

cpp

clog << "This is a log message." << endl;

**Summary**

C++ provides robust input and output capabilities through various header files:

* <iostream> handles console I/O.
* <iomanip> formats input and output.
* <fstream> manages file I/O.