Formatted I/O operations in C++ allow you to control the way input and output are presented, which is particularly useful when you need data to be displayed or read in a specific format. C++ provides a variety of ways to perform formatted I/O using streams (cin, cout, cerr, etc.), manipulators, and functions.

### **Basic Stream I/O**

* **cout**: Used for output operations.
* **cin**: Used for input operations.
* **cerr**: Used for error output.

### **Manipulators**

Manipulators are used to format the output or input. Here are some common manipulators:

1. **std::setw(n)**: Sets the width of the next input/output field.
2. **std::setfill(c)**: Sets the fill character to c.
3. **std::setprecision(n)**: Sets the decimal precision for floating-point numbers.
4. **std::fixed and std::scientific**: Forces the output of floating-point numbers to be in fixed-point notation or scientific notation.
5. **std::left, std::right, std::internal**: Aligns the output to the left, right, or internal (for signs) within the field width.

### **Examples of Formatted I/O Operations**

#### **Example 1: Setting Width and Filling with Characters**

#include <iostream>

#include <iomanip>

int main() {

std::cout << std::setw(10) << 42 << std::endl; // Output: " 42"

std::cout << std::setw(10) << std::setfill('\*') << 42 << std::endl; // Output: "\*\*\*\*\*\*\*\*42"

return 0;

}

#### 

#### 

#### **Example 2: Setting Precision for Floating-Point Numbers**

#include <iostream>

#include <iomanip>

int main() {

double pi = 3.141592653589793;

std::cout << std::setprecision(2) << pi << std::endl; // Output: 3.1

std::cout << std::setprecision(5) << pi << std::endl; // Output: 3.1416

std::cout << std::fixed << std::setprecision(3) << pi << std::endl; // Output: 3.142

std::cout << std::scientific << std::setprecision(3) << pi << std::endl; // Output: 3.142e+00

return 0;

}

#### **Example 3: Aligning Output**

#include <iostream>

#include <iomanip>

int main() {

std::cout << std::left << std::setw(10) << 42 << std::endl; // Output: "42 "

std::cout << std::right << std::setw(10) << 42 << std::endl; // Output: " 42"

std::cout << std::internal << std::setw(10) << -42 << std::endl; // Output: "- 42"

return 0;

}

### **I/O with Strings**

Formatted I/O operations can also be performed with strings. You can use std::getline() for reading strings with spaces, and manipulators to control how strings are displayed.

#### **Example 4: Reading and Writing Strings**

#include <iostream>

#include <iomanip>

#include <string>

int main() {

std::string name;

std::cout << "Enter your name: ";

std::getline(std::cin, name);

std::cout << "Hello, " << std::setw(20) << std::left << name << "!" << std::endl;

return 0;

}

### **Summary**

* **std::cout** and **std::cin** are basic I/O streams for output and input.
* Manipulators like **std::setw**, **std::setfill**, and **std::setprecision** help in formatting the data.
* **Alignment** (std::left, std::right, std::internal) can be used to control the position of data in the output.