

Introduction

A digital computer is a useful tool for solving a great variety of **problems**. A solution to a problem is called an **algorithm**; it describes the sequence of steps to be performed for the problem to be solved. High-level languages such as C++ provide a more convenient notation for implementing algorithms.

A Simple C++ Program

1. Outputs the message Hello World.

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

int main(void)
{
    cout << "Hello World\n";
    return 0;
}
```

#include is **preprocessor directive** used to include the contents of the header file `iostream` in the program. `iostream` is a standard C++ header file and contains definitions for input and output.

main is a function may have zero or more **parameters**; these always appear after the function name, between a pair of brackets. The word `void` appearing between the brackets indicates that `main` has no parameters. A function may also have a **return type**; this always appears before the function name. The return type for `main` is `int` (i.e., an integer number). All C++ programs must have exactly one `main` function. Program execution always begins from `main`.

A statement is a computation step which may produce a value. The end of a statement is always marked with a semicolon (;). This statement causes the **string** `"Hello World\n"` to be sent to the `cout` **output stream**. A string is any sequence of characters enclosed in double-quotes. The last character in this string (`\n`) is a newline character.

2. Program illustrates the uses of some simple variable:

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

int main(void)
{
    int workDays;
    float workHours, payRate, weeklyPay;
    workDays = 5;
    workHours = 7.5;
    payRate = 38.55;
    weeklyPay = workDays * workHours * payRate;
    cout << "Weekly Pay = ";
    cout << weeklyPay;
    cout << '\n';

    return 0;
}
```

Simple Input/Output

The most common way in which a program communicates with the outside world is through simple, character-oriented Input/Output (IO) operations. C++ provides two useful operators for this purpose: >> for input and << for output. We have already seen examples of output using <<.

3. Program illustrates the uses of some simple input/output:

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

int main(void)
{
    int workDays = 5;
    float workHours = 7.5;
    float payRate, weeklyPay;
    cout << "What is the hourly pay rate? ";
    cin >> payRate;
    weeklyPay = workDays * workHours * payRate;
    cout << "Weekly Pay = ";
    cout << weeklyPay;
    cout << '\n';

    return 0;
}
```

Comments

A comment is a piece of descriptive text which explains some aspect of a program. Program comments are totally ignored by the compiler and are only intended for human readers. C++ provides two types of comment delimiters:

- Anything after `//` (until the end of the line on which it appears) is considered a comment.
- Anything enclosed by the pair `/*` and `*/` is considered a comment.

```
#include <iostream>
using namespace std;
/* This program calculates the weekly gross pay for a worker,
based on the total number of hours worked and the hourly pay
rate. */
int main(void)
{
    int workDays = 5; // Number of work days per week
    float workHours = 7.5; // Number of work hours per day
    float payRate = 33.50; // Hourly pay rate
    float weeklyPay; // Gross weekly pay
    weeklyPay = workDays * workHours * payRate;
    cout << "Weekly Pay = " << weeklyPay << '\n';

    return 0;
}
```

4. Program to test arithmetic operators.

```
#include <iostream>
using namespace std;
int main(void)
{
    int num1, num2;
    cout << "Enter two numbers to be operated with arithmetic
operators : ";
    cin >> num1 >> num2;
    cout << endl;
    cout << num1 << "+" << num2 << "=" << num1 + num2 << endl;
    cout << num1 << "*" << num2 << "=" << num1 * num2 << endl;
    cout << num1 << "-" << num2 << "=" << num1 - num2 << endl;
    cout << num1 << "/" << num2 << "=" << num1 / num2 << endl;
    return 0;
}
```

Variables

While writing program in any language, you need to use various variables to store various information. Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

| Type | Keyword |
|-----------------------|---------|
| Boolean | bool |
| Character | char |
| Integer | int |
| Floating point | float |
| Double floating point | double |

An **integer variable** may be defined to be of type `short`, `int`, or `long`. The only difference is that an `int` uses more or at least the same number of bytes as a `short`, and a `long` uses more or at least the same number of bytes as an `int`. For example, on the author's PC, a `short` uses 2 bytes, an `int` also 2 bytes, and a `long` 4 bytes.

```
short age = 20;
int salary = 65000;
long price = 4500000;
```

By default, an integer variable is assumed to be signed (i.e., have a signed representation so that it can assume positive as well as negative values). However, an integer can be defined to be unsigned by using the keyword `unsigned` in its definition. The keyword `signed` is also allowed but is redundant.

```
unsigned short age = 20;
unsigned int salary = 65000;
unsigned long price = 4500000;
```

A **real variable** may be defined to be of type `float` or `double`. The latter uses more bytes and therefore offers a greater range and accuracy for representing real numbers. For example, on the author's PC, a `float` uses 4 and a `double` uses 8 bytes.

```
float interestRate = 0.06;
double pi = 3.141592654;
```

A **character variable** is defined to be of type `char`. A character variable occupies a single byte which contains the *code* for the character. This code is a numeric value and depends on the *character coding system* being used (i.e., is machine-dependent). The most common system is ASCII (American Standard Code for

Information Interchange). For example, the character *A* has the ASCII code 65, and the character *a* has the ASCII code 97.

```
char ch = 'A';
```

A string is a consecutive sequence (i.e., *array*) of characters which are terminated by a null character. A **string variable** is defined to be of type `char*`

```
char *str = "HELLO";
```

5. Program compute area and circumference of a circle

```
#include <iostream>
using namespace std;
int main(void)
{
    float radius, area, circumference;
    float pi = 3.14;
    cout << "Enter radius: ";
    cin >> radius;
    area = pi * radius * radius;
    circumference = 2 * pi * radius;
    cout << "Area = " << area << endl;
    cout << "Circumference = " << circumference << endl;
    return 0;
}
```

Exercises

1. Write a program which inputs a temperature reading expressed in Fahrenheit and outputs its equivalent in Celsius, using the formula:

$$^{\circ}\text{C} = \frac{5}{9} (^{\circ}\text{F} - 32)$$

Compile and run the program. Its behavior should resemble this:

```
Temperature in Fahrenheit: 41
41 degrees Fahrenheit = 5 degrees Celsius
```