Meaning

**C++** is an object-oriented **programming** (OOP) **language** that is viewed by many as the best **language** for creating large-scale applications. **C++** is a superset of the C **language**. A related **programming language**, Java, is based on **C++** but optimized for the distribution of program objects in a network such as the Internet.

Syntax

C++ Program Structure

* The C++ language defines several headers, which contain information that is either necessary or useful to your program. For this program, the header **<iostream>** is needed.
* The line **using namespace std;** tells the compiler to use the std namespace. Namespaces are a relatively recent addition to C++.
* The next line '**// main() is where program execution begins.**' is a single-line comment available in C++. Single-line comments begin with // and stop at the end of the line.
* The line **int main()** is the main function where program execution begins.
* The next line **cout << "Hello World";** causes the message "Hello World" to be displayed on the screen.
* The next line **return 0;** terminates main( )function and causes it to return the value 0 to the calling process.

## Compile and Execute C++ Program

* Open a text editor and add the code as above.
* Save the file as: hello.cpp
* Open a command prompt and go to the directory where you saved the file.
* Type 'g++ hello.cpp' and press enter to compile your code. If there are no errors in your code the command prompt will take you to the next line and would generate a.out executable file.
* Now, type 'a.out' to run your program.
* You will be able to see ' Hello World ' printed on the window.

Conditional Statement

C++ Conditions and If Statements

C++ supports the usual logical conditions from mathematics:

* Less than: a < b
* Less than or equal to: a <= b
* Greater than: a > b
* Greater than or equal to: a >= b
* Equal to a == b
* Not Equal to: a != b

You can use these conditions to perform different actions for different decisions.

C++ has the following conditional statements:

* Use if to specify a block of code to be executed, if a specified condition is true
* Use else to specify a block of code to be executed, if the same condition is false
* Use else if to specify a new condition to test, if the first condition is false
* Use switch to specify many alternative blocks of code to be executed

Note that if is in lowercase letters. Uppercase letters (If or IF) will generate an error.

The if statement

Use the if statement to specify a block of C++ code to be executed if a condition is true.

Syntax:

if (condition) {  
  *// block of code to be executed if the condition is true*  
}

## The else Statement

Use the else statement to specify a block of code to be executed if the condition is false.

Syntax

if (condition) {  
  *// block of code to be executed if the condition is true*  
} else {  
  *// block of code to be executed if the condition is false*  
}

## The else if Statement

Use the else if statement to specify a new condition if the first condition is false.

Syntax

if (condition1) {  
  *// block of code to be executed if condition1 is true*  
} else if (condition2) {  
  *// block of code to be executed if the condition1 is false and condition2 is true*  
} else {  
  *// block of code to be executed if the condition1 is false and condition2 is false*  
}

## C++ Switch Statements

Use the switch statement to select one of many code blocks to be executed.

Syntax

switch(expression) {  
  case x:  
    *// code block*  
    break;  
  case y:  
    *// code block*  
    break;  
  default:  
    *// code block*  
}

This is how it works:

* The switch expression is evaluated once
* The value of the expression is compared with the values of each case
* If there is a match, the associated block of code is executed
* The break and default keywords are optional, and will be described later in this chapter

## C++ While Loop

The while loop loops through a block of code as long as a specified condition is true:

Syntax

while (condition) {  
*// code block to be executed*  
}

## The Do/While Loop

The do/while loop is a variant of the while loop. This loop will execute the code block once, before checking if the condition is true, then it will repeat the loop as long as the condition is true.

Syntax

do {  
*// code block to be executed*}  
while (condition);

## C++ For Loop

When you know exactly how many times you want to loop through a block of code, use the for loop instead of a while loop:

Syntax

for (*statement 1*;*statement 2*;*statement 3*) {  
  *// code block to be executed*  
}

**Statement 1** is executed (one time) before the execution of the code block.

**Statement 2** defines the condition for executing the code block.

**Statement 3** is executed (every time) after the code block has been executed.