

# Hello World Program

```
// Hello world program in C++

#include<iostream>
using namespace std;

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

## 1. Comments

The two slash(//) signs are used to add comments in a program. It does not have any effect on the behaviour or outcome of the program. It is used to give description of the program you're writing.

## 2. #include<iostream>

#include is the pre-processor directive that is used to include files in our program. Here we are including the iostream standard file which is necessary for the declarations of basic standard input/output library in C++.

## 3. Using namespace std

All elements of the standard C++ library are declared within namespace. Here we are using std namespace.

## 4. int main()

The execution of any C++ program starts with the main function, hence it is necessary to have a main function in your program. 'int' is the return value of this function. (We will be studying about functions in more detail later).

## 5. {}

The curly brackets are used to indicate the starting and ending point of any function. Every opening bracket should have a corresponding closing bracket.

6. `cout<<"Hello World!\n";`

This is a C++ statement. **cout** represents the standard output stream in C++. It is declared in the `iostream` standard file within the `std` namespace. The text between quotations will be printed on the screen.

`\n` will not be printed, it is used to add line break.

Each statement in C++ ends with a semicolon (`;`)

7. `return 0;`

`return` signifies the end of a function. Here the function is `main`, so when we hit `return 0`, it exits the program. We are returning 0 because we mentioned the return type of `main` function as integer (`int main`). A zero indicates that everything went fine and a one indicates that something has gone wrong.

## Input and Output in C++

The header file `iostream` must be included to make use of the input/output (`cin/cout`) operators.

### Standard Output (`cout`)

By default, the standard output of a program points at the screen. So with the `cout` operator and the "insertion" operator (`<<`) you can print a message onto the screen.

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

int main()
{
    cout << "Hello World!";
    return 0;
}
```

To print the content of a variable the double quotes are not used.

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    char Yes = 'y';
```

```
    cout << Yes;
```

```
    return 0;
```

```
}
```

The << operator can be used multiple times in a single statement. Take a look at an example:

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    cout << "Hello, " << "this is a test " << "string.";
```

```
    return 0;
```

```
}
```

It is possible to combine variables and text:

```
#include<iostream>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    char Yes = 'y';
```

```
    cout << "Print the character " << Yes;
```

```
    return 0;
```

```
}
```

The cout operator does not put a line break at the end of the output. So if you want to print two sentences you will have to use the new-line character ( \n ).

```
cout << "This is one sentence.\n";  
cout << "This is another.\n";
```

It is possible to use the **endl** manipulator instead of the new-line character.

```
cout << "This is one sentence." << endl;  
cout << "This is another." << endl;
```

## Standard input (cin)

In most cases the standard input device is the keyboard. With the cin and >> operators it is possible to read input from the keyboard.

Take a look at an example:

```
#include<iostream>  
using namespace std;  
  
int main()  
{  
    char ch;  
    cout << "Press a character and press return: ";  
    cin >> ch;  
    cout << ch;  
    return 0;  
}
```

```
}
```

**Note:** The input is processed by cin after the return key is pressed.

The cin operator will always return the variable type that you use with cin. So if you request an integer you will get an integer and so on. This can cause an error when the user of the program does not return the type that you are expecting. (Example: you ask for an integer and you get a string of characters.)

The cin operator is also chainable. For example:

```
cin >> X >> Y;
```

In this case the user must give two input values, that are separated by any valid blank separator (tab, space or new-line).