

C++ if, if...else and Nested if...else

 programiz.com/cpp-programming/if-else

Join our newsletter for the latest updates.

In this tutorial, we will learn about the if...else statement to create decision making programs with the help of examples.

In computer programming, we use the `if` statement to run a block code only when a certain condition is met.

For example, assigning grades (A, B, C) based on marks obtained by a student.

- if the percentage is above **90**, assign grade **A**
- if the percentage is above **75**, assign grade **B**
- if the percentage is above **65**, assign grade **C**

There are three forms of `if...else` statements in C++.

1. `if` statement
2. `if...else` statement
3. `if...else if...else` statement

C++ if Statement

The syntax of the `if` statement is:

```
if (condition) {  
    // body of if statement  
}
```

The `if` statement evaluates the `condition` inside the parentheses `()`.

- If the `condition` evaluates to `true`, the code inside the body of `if` is executed.
- If the `condition` evaluates to `false`, the code inside the body of `if` is skipped.

Note: The code inside `{ }` is the body of the `if` statement.

Condition is true

```
int number = 5;

if (number > 0) {
    // code
}

// code after if
```

Condition is false

```
int number = 5;

if (number < 0) {
    // code
}

// code after if
```

Working of C++ if Statement

Example 1: C++ if Statement

```
// Program to print positive number entered by the user
// If the user enters a negative number, it is skipped

#include <iostream>
using namespace std;

int main() {
    int number;

    cout << "Enter an integer: ";
    cin >> number;

    // checks if the number is positive
    if (number > 0) {
        cout << "You entered a positive integer: " << number << endl;
    }
    cout << "This statement is always executed.";
    return 0;
}
```

Output 1

```
Enter an integer: 5
You entered a positive number: 5
This statement is always executed.
```

When the user enters `5`, the condition `number > 0` is evaluated to `true` and the statement inside the body of `if` is executed.

Output 2

```
Enter a number: -5
This statement is always executed.
```

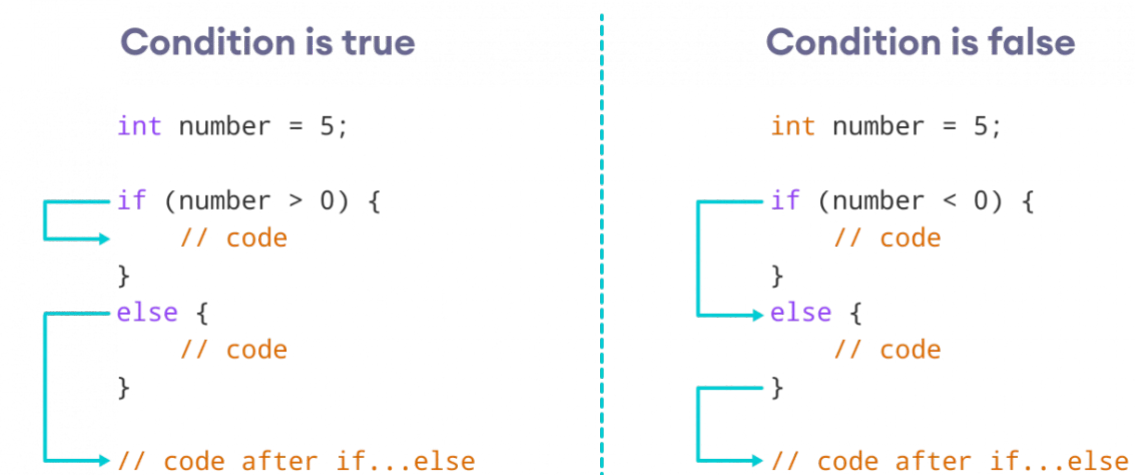
When the user enters `-5`, the condition `number > 0` is evaluated to `false` and the statement inside the body of `if` is not executed.

C++ if...else

The `if` statement can have an optional `else` clause. Its syntax is:

```
if (condition) {  
    // block of code if condition is true  
}  
else {  
    // block of code if condition is false  
}
```

The `if...else` statement evaluates the `condition` inside the parenthesis.



Working of C++ if...else

If the `condition` evaluates `true`,

- the code inside the body of `if` is executed
- the code inside the body of `else` is skipped from execution

If the `condition` evaluates `false`,

- the code inside the body of `else` is executed
- the code inside the body of `if` is skipped from execution

Example 2: C++ if...else Statement

```
// Program to check whether an integer is positive or negative
// This program considers 0 as a positive number

#include <iostream>
using namespace std;

int main() {
    int number;

    cout << "Enter an integer: ";
    cin >> number;
    if (number >= 0) {
        cout << "You entered a positive integer: " << number << endl;
    }
    else {
        cout << "You entered a negative integer: " << number << endl;
    }
    cout << "This line is always printed.";
    return 0;
}
```

Output 1

```
Enter an integer: 4
You entered a positive integer: 4.
This line is always printed.
```

In the above program, we have the condition `number >= 0` . If we enter the number greater or equal to 0, then the condition evaluates `true` .

Here, we enter 4. So, the condition is `true` . Hence, the statement inside the body of `if` is executed.

Output 2

```
Enter an integer: -4
You entered a negative integer: -4.
This line is always printed.
```

Here, we enter -4. So, the condition is `false` . Hence, the statement inside the body of `else` is executed.

C++ if...else...else if statement

The `if...else` statement is used to execute a block of code among two alternatives. However, if we need to make a choice between more than two alternatives, we use the `if...else if...else` statement.

The syntax of the `if...else if...else` statement is:

```

if (condition1) {
    // code block 1
}
else if (condition2){
    // code block 2
}
else {
    // code block 3
}

```

Here,

- If `condition1` evaluates to `true`, the `code block 1` is executed.
- If `condition1` evaluates to `false`, then `condition2` is evaluated.
- If `condition2` is `true`, the `code block 2` is executed.
- If `condition2` is `false`, the `code block 3` is executed.

1st Condition is true

```

int number = 2;
if (number > 0) {
    // code
}
else if (number == 0){
    // code
}
else {
    //code
}
//code after if

```

2nd Condition is true

```

int number = 0;
if (number > 0) {
    // code
}
else if (number == 0){
    // code
}
else {
    //code
}
//code after if

```

All Conditions are false

```

int number = -2;
if (number > 0) {
    // code
}
else if (number == 0){
    // code
}
else {
    //code
}
//code after if

```

How if...else if...else Statement Works

Note: There can be more than one `else if` statement but only one `if` and `else` statements.

Example 3: C++ if...else...else if

```
// Program to check whether an integer is positive, negative or zero

#include <iostream>
using namespace std;

int main() {
    int number;

    cout << "Enter an integer: ";
    cin >> number;
    if (number > 0) {
        cout << "You entered a positive integer: " << number << endl;
    }
    else if (number < 0) {
        cout << "You entered a negative integer: " << number << endl;
    }
    else {
        cout << "You entered 0." << endl;
    }
    cout << "This line is always printed.";
    return 0;
}
```

Output 1

```
Enter an integer: 1
You entered a positive integer: 1.
This line is always printed.
```

Output 2

```
Enter an integer: -2
You entered a negative integer: -2.
This line is always printed.
```

Output 3

```
Enter an integer: 0
You entered 0.
This line is always printed.
```

In this program, we take a number from the user. We then use the `if...else if...else` ladder to check whether the number is positive, negative, or zero.

If the number is greater than `0`, the code inside the `if` block is executed. If the number is less than `0`, the code inside the `else if` block is executed. Otherwise, the code inside the `else` block is executed.

C++ Nested if...else

Sometimes, we need to use an `if` statement inside another `if` statement. This is known as nested `if` statement.

Think of it as multiple layers of `if` statements. There is a first, outer `if` statement, and inside it is another, inner `if` statement. Its syntax is:

```
// outer if statement
if (condition1) {
    // statements

    // inner if statement
    if (condition2) {
        // statements
    }
}
```

Notes:

- We can add `else` and `else if` statements to the inner `if` statement as required.
- The inner `if` statement can also be inserted inside the outer `else` or `else if` statements (if they exist).
- We can nest multiple layers of `if` statements.

Example 4: C++ Nested if

```
// C++ program to find if an integer is even or odd or neither (0)
// using nested if statements

#include <iostream>
using namespace std;

int main() {
    int num;

    cout << "Enter an integer: ";
    cin >> num;

    // outer if condition
    if (num != 0) {

        // inner if condition
        if ((num % 2) == 0) {
            cout << "The number is even." << endl;
        }
        // inner else condition
        else {
            cout << "The number is odd." << endl;
        }
    }
    // outer else condition
    else {
        cout << "The number is 0 and it is neither even nor odd." << endl;
    }
    cout << "This line is always printed." << endl;
}
```

Output 1

```
Enter an integer: 34
The number is even.
This line is always printed.
```

Output 2

```
Enter an integer: 35
The number is odd.
This line is always printed.
```

Output 3

```
Enter an integer: 0
The number is 0 and it is neither even nor odd.
This line is always printed.
```

In the above example,

- We take an integer as an input from the user and store it in the variable *num*.
- We then use an `if...else` statement to check whether *num* is not equal to `0`.
 - If `true`, then the **inner** `if...else` statement is executed.
 - If `false`, the code inside the **outer** `else` condition is executed, which prints "The number is 0 and neither even nor odd."
- The **inner** `if...else` statement checks whether the input number is divisible by `2`.
 - If `true`, then we print a statement saying that the number is even.
 - If `false`, we print that the number is odd.

Notice that `0` is also divisible by `2`, but it is actually not an even number. This is why we first make sure that the input number is not `0` in the outer `if` condition.

Note: As you can see, nested `if...else` makes your logic complicated. If possible, you should always try to avoid nested `if...else`.

Body of if...else With Only One Statement

If the body of `if...else` has only one statement, you can omit `{ }` in the program. For example, you can replace

```
int number = 5;

if (number > 0) {
    cout << "The number is positive." << endl;
}
else {
    cout << "The number is negative." << endl;
}
```

with


```
int number = 5;

if (number > 0)
    cout << "The number is positive." << endl;
else
    cout << "The number is negative." << endl;
```

The output of both programs will be the same.

Note: Although it's not necessary to use `{ }` if the body of `if...else` has only one statement, using `{ }` makes your code more readable.

More on Decision Making

In certain situations, a **ternary operator** can replace an `if...else` statement. To learn more, visit [C++ Ternary Operator](#).

If we need to make a choice between more than one alternatives based on a given test condition, the `switch` statement can be used. To learn more, visit [C++ switch](#).

Check out these examples to learn more:

[C++ Program to Check Whether Number is Even or Odd](#)

[C++ Program to Check Whether a Character is Vowel or Consonant](#)

[C++ Program to Find Largest Number Among Three Numbers](#)