C - The If Statement

Conditional execution of instructions is the basic requirement of a computer program. The **if** statement in C is the primary conditional statement. C allows an optional **else** keyword to specify the statements to be executed if the **if** condition is false.

C - if Statement

The **if** statement is a fundamental decision control statement in C programming. One or more statements in a block will get executed depending on whether the Boolean condition in the **if** statement is true or false.

Syntax of if Statement

The **if** statement is written with the following syntax –

```
if(boolean_expression) {
   /* statement(s) will execute if the boolean expression is true */
}
```


How if Statement Works?

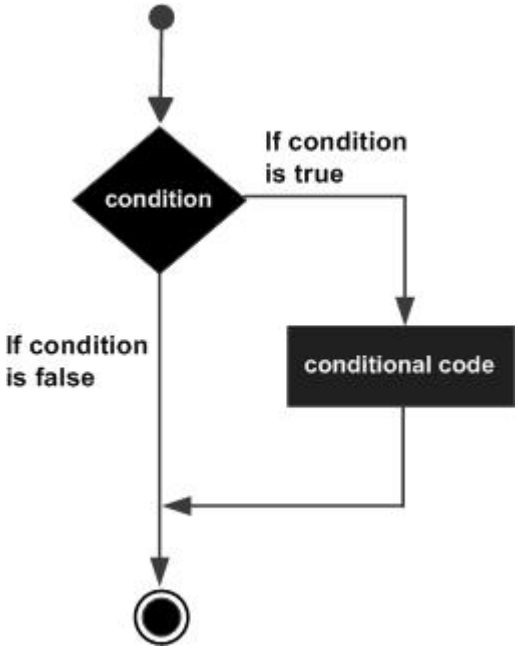
C uses a pair of curly brackets to form a code block. If the Boolean expression evaluates to true, then the block of code inside the **if** statement will be executed.

If the Boolean expression evaluates to false, then the first set of code after the end of the **if** statement (after the closing curly brace) will be executed.

C programming treats any non-zero and non-null values as true. And if the values are either zero or null, then they are treated as false values.

Flowchart of if Statement

The behaviour of the **if** statement is expressed by the following flowchart –



Flowchart Explained

When the program control comes across the **if** statement, the condition is evaluated.

If the condition is true, the statements inside the **if** block are executed.

If the condition is false, the program flow bypasses the conditional block.

Statements after the **if** block are executed to continue the program flow.

Example of if Statement in C

This example demonstrates the simplest use-case of **if** statement. It determines and tells the user if the value of a variable is less than 20.

</> Open Compiler

```
#include <stdio.h>

int main (){

   /* local variable declaration */
   int a;

   // run the program for different values of "a"
   // Assign 12 first and 40 afterwards

   a = 12;  //change to 40 and run again
   printf("Value of a is : %d\n", a);
```

```
   // check the boolean condition using if statement

   if(a < 20){
      //if the condition is true, then print the following
      printf("a is less than 20\n" );
   }
   return 0;
}
```

Output

Run the above program and check its ouput –

```
Value of a is : 12
a is less than 20
```

Now assign a number greater than 20. The **if** condition is not executed.

```
Value of a is: 40
```

if Statement with Logical Operations

You can put a compound boolean expression with the use of **&&** or **||** operators in the parenthesis in the **if** statement.

Example

In the following example, three variables "a", "b" and "c" are compared. The **if** block will be executed when "a" is greater than both "b" and "c".

</> Open Compiler

```
#include <stdio.h>

int main () {

   /* local variable declaration */
   int a, b, c;

   /*use different values for a, b and c as
   10, 5, 7
```

```
   10, 20, 15
   */

   // change to 10,20,15 respectively next time
   a = 10; b = 5; c = 7;

   if (a>=b && a>=c){
      printf ("a is greater than b and c \n");
   }
   printf("a: %d b:%d c:%d", a, b, c);

   return 0;
}
```

Output

Run the code and check its output –

```
//when values for a, b and c are 10 5 7
a is greater than b and c
a: 10 b:5 c:7

//when values for a, b and c are 10 20 15
a: 10 b:20 c:15
```

Note that the statement following the conditional block is executed after the block is executed. If the condition is false, the program jumps directly to the statement after the block.

Multiple if Statements

If you have multiple conditions to check, you can use the if statement multiple times.

Example

In this example, the net payable amount is calculated by applying discount on the bill amount.

The discount applicable is 5 percent if the amount is between 1000 to 5000, and 10 percent if the amount is above 5000. No discount is applicable for purchases below 1000.

</> Open Compiler

```c
#include <stdio.h>

int main () {

   // local variable declaration
   int amount;
   float discount, net;

   /*Run the program for different values
   of amount - 500, 2250 and 5200. Blocks in
   respective conditions will be executed*/

   // change to 2250 and 5200 and run again
   amount = 500;

   if (amount < 1000){
      discount=0;
   }
   if (amount >= 1000 && amount<5000){
      discount=5;
   }
   if (amount >= 5000){
      discount=10;
   }
   net = amount - amount*discount/100;
   printf("Amount: %d Discount: %f Net payable: %f", amount, discount, net);

   return 0;
}
```

Output

```
//when the bill amount is 500
Amount: 500 Discount: 0.000000 Net payable: 500.000000

//when the bill amount is 2250
Amount: 2250 Discount: 5.000000 Net payable: 2137.500000

//when the bill amount is 5200
Amount: 5200 Discount: 10.000000 Net payable: 4680.000000
```

Checking Multiple Conditions With if Statement

You can also check multiple conditions using the logical operators in a single if statement.

Example

In this program, a student is declared as **passed** only if the average of "phy" and "maths" marks is greater than equal to 50. Also, the student should have secured more than 35 marks in both the subjects. Otherwise, the student is declared as **failed**.

</> Open Compiler

```
#include <stdio.h>

int main (){
   /* local variable declaration */
   int phy, maths;
   float avg;

   /*use different values of phy and maths
   to check conditional execution*/

   //change to 40, 40 and 80, 40
   phy = 50; maths = 50;
   avg = (float)(phy + maths)/2;
   printf("Phy: %d Maths: %d Avg: %f\n", phy, maths, avg);

   if (avg >= 50 && (maths >= 35 && phy >= 35)){
      printf("Result: Pass");
   }

   if (avg<50) {
      printf("Result: Fail\n");
   }
   return 0;
}
```

Output

Run the code and check its output –

```
//when marks in Phy and Maths - 50 50
Phy: 50 Maths: 50 Avg: 50.000000
Result: Pass

//when marks in Phy and Maths - 40 40
Phy: 40 Maths: 40 Avg: 40.000000
Result: Fail

//when marks in Phy and Maths - 80 40
Phy: 80 Maths: 40 Avg: 60.000000
Result: Pass
```