#  C# - Decision Making

Decision making structures requires the programmer to specify one or more conditions to be evaluated or tested by the program, along with a statement or statements to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.

Following is the general form of a typical decision making structure found in most of the programming languages −



C# provides following types of decision making statements. Click the following links to check their detail.

An **if** statement consists of a boolean expression followed by one or more statements.

## Syntax

The syntax of an if statement in C# is −

if(boolean\_expression) {

 /\* statement(s) will execute if the boolean expression is true \*/

}

If the boolean expression evaluates to **true**, then the block of code inside the if statement is executed. If boolean expression evaluates to **false**, then the first set of code after the end of the if statement(after the closing curly brace) is executed.

## Flow Diagram



## Example

[Live Demo](http://tpcg.io/07w42D)

using System;

namespace DecisionMaking {

 class Program {

 static void Main(string[] args) {

 /\* local variable definition \*/

 int a = 10;

 /\* check the boolean condition using if statement \*/

 if (a < 20) {

 /\* if condition is true then print the following \*/

 Console.WriteLine("a is less than 20");

 }

 Console.WriteLine("value of a is : {0}", a);

 Console.ReadLine();

 }

 }

}

When the above code is compiled and executed, it produces the following result −

a is less than 20;

value of a is : 10

An **if** statement can be followed by an optional **else** statement, which executes when the boolean expression is false.

## Syntax

The syntax of an **if...else** statement in C# is −

if(boolean\_expression) {

 /\* statement(s) will execute if the boolean expression is true \*/

} else {

 /\* statement(s) will execute if the boolean expression is false \*/

}

If the boolean expression evaluates to **true**, then the **if block** of code is executed, otherwise **else block** of code is executed.

## Flow Diagram



## Example

[Live Demo](http://tpcg.io/DeLC15)

using System;

namespace DecisionMaking {

 class Program {

 static void Main(string[] args) {

 /\* local variable definition \*/

 int a = 100;

 /\* check the boolean condition \*/

 if (a < 20) {

 /\* if condition is true then print the following \*/

 Console.WriteLine("a is less than 20");

 } else {

 /\* if condition is false then print the following \*/

 Console.WriteLine("a is not less than 20");

 }

 Console.WriteLine("value of a is : {0}", a);

 Console.ReadLine();

 }

 }

}

When the above code is compiled and executed, it produces the following result −

a is not less than 20;

value of a is : 100

## The if...else if...else Statement

An **if** statement can be followed by an optional **else if...else** statement, which is very useful to test various conditions using single if...else if statement.

When using if, else if, else statements there are few points to keep in mind.

* An if can have zero or one else's and it must come after any else if's.
* An if can have zero to many else if's and they must come before the else.
* Once an else if succeeds, none of the remaining else if's or else's will be tested.

## Syntax

The syntax of an **if...else if...else** statement in C# is −

if(boolean\_expression 1) {

 /\* Executes when the boolean expression 1 is true \*/

}

else if( boolean\_expression 2) {

 /\* Executes when the boolean expression 2 is true \*/

}

else if( boolean\_expression 3) {

 /\* Executes when the boolean expression 3 is true \*/

} else {

 /\* executes when the none of the above condition is true \*/

}

## Example

[Live Demo](http://tpcg.io/9U1yTi)

using System;

namespace DecisionMaking {

 class Program {

 static void Main(string[] args) {

 /\* local variable definition \*/

 int a = 100;

 /\* check the boolean condition \*/

 if (a == 10) {

 /\* if condition is true then print the following \*/

 Console.WriteLine("Value of a is 10");

 }

 else if (a == 20) {

 /\* if else if condition is true \*/

 Console.WriteLine("Value of a is 20");

 }

 else if (a == 30) {

 /\* if else if condition is true \*/

 Console.WriteLine("Value of a is 30");

 } else {

 /\* if none of the conditions is true \*/

 Console.WriteLine("None of the values is matching");

 }

 Console.WriteLine("Exact value of a is: {0}", a);

 Console.ReadLine();

 }

 }

}

When the above code is compiled and executed, it produces the following result −

None of the values is matching

Exact value of a is: 100