**C# Expressions, Statements and Blocks (With Examples)**

## C# Expressions

An expression in C# is a combination of operands (variables, literals, method calls) and operators that can be evaluated to a single value. To be precise, an expression must have at least one operand but may not have any operator.

Let's look at the example below:

double temperature;

temperature = 42.05;

Here, 42.05 is an expression. Also, temperature = 42.05 is an expression too.

int a, b, c, sum;

sum = a + b + c;

Here, a + b + c is an expression.

if (age>=18 && age<58)

Console.WriteLine("Eligible to work");

Here, (age>=18 && age<58) is an expression that returns a boolean value. "Eligible to work" is also an expression.

## C# Statements

A statement is a basic unit of execution of a program. A program consists of multiple statements.

For example:

int age = 21;

Int marks = 90;

In the above example, both lines above are statements.

There are different types of statements in C#. In this tutorial, we’ll mainly focus on two of them:

1. Declaration Statement
2. Expression Statement

### Declaration Statement

Declaration statements are used to declare and initialize variables.

For example:

char ch;

int maxValue = 55;

Both char ch; and int maxValue = 55; are declaration statements.

### Expression Statement

An expression followed by a semicolon is called an expression statement.

For example:

/\* Assignment \*/

area = 3.14 \* radius \* radius;

/\* Method call is an expression\*/

System.Console.WriteLine("Hello");

Here, 3.14 \* radius \* radius  is an expression and area = 3.14 \* radius \* radius; is an expression statement.

Likewise, System.Console.WriteLine("Hello"); is both an expression and a statement.

Beside declaration and expression statement, there are:

* Selection Statements (if...else, switch)
* Iteration Statements (do, while, for, foreach)
* Jump Statements (break, continue, goto, return, yield)
* Exception Handling Statements (throw, try-catch, try-finally, try-catch-finally)

These statements will be discussed in later tutorials.

If you want to learn more about statements, visit [C# Statements](https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/statements-expressions-operators/statements) ( C# reference)

## C# Blocks

A block is a combination of zero or more statements that is enclosed inside curly brackets { }.

For example:

## Example 1: C# Blocks with statements

using System;

namespace Blocks

{

class BlockExample

{

public static void Main(string[] args)

{

double temperature = 42.05;

if (temperature > 32)

{ // Start of block

Console.WriteLine("Current temperature = {0}", temperature);

Console.WriteLine("It's hot");

} // End of block

}

}

}

When we run the program, the output will be:

Current temperature = 42.05

It's hot

Here, the two statements inside { }:

Console.WriteLine("Current temperature = {0}", temperature);

and

Console.WriteLine("It's hot");

forms a **block**.

### Example 2: C# Blocks without statements

A block may not have any statements within it as shown in the below example.

using System;

namespace Blocks

{

class BlockExample

{

public static void Main(string[] args)

{

double temperature = 42.05;

if (temperature > 32)

{ // Start of block

// No statements

} // End of block

}

}

}

Here, the curly braces { } after if(temperature > 32) contains only comments and no statements.