**C# continue Statement**

In this tutorial, you will learn about the working of C# continue statement with the help of examples.

In C#, we use the continue statement to skip a current iteration of a loop.

When our program encounters the continue statement, the program control moves to the end of the loop and executes the test condition (update statement in case of for loop).

The syntax for continue is:

continue;

Before we learn about continue, make sure to learn about

* [for loop](https://www.programiz.com/csharp-programming/for-loop)
* [while loop](https://www.programiz.com/csharp-programming/do-while-loop)
* [if...else](https://www.programiz.com/csharp-programming/if-else-statement)

**Example1: C# continue with for loop**

using System;

namespace ContinueLoop {

class Program {

static void Main(string[] args){

for (int i = 1; i <= 5; ++i{

if (i == 3) {

continue;

}

Console.WriteLine(i);

}

}

}

}

**Output**

1

2

4

5

In the above example, we have used the for loop to print numbers from i = 1 to 5. However, the number **3** is not printed.

This is because when the value of i is **3**, the continue statement is executed.

// skips the condition

if (i == 3) {

continue;

}

This skips the current iteration of loop and moves the program control to the update statement. Hence, the value **3** is not printed.

**Note**: The continue statement is usually used with the if...else statement.

**Example: C# continue with while loop**

using System;

namespace ContinueWhile {

class Program{

static void Main(string[] args) {

int i = 0;

while (i < 5) {

i++;

if (i == 3) {

continue;

}

Console.WriteLine(i);

}

}

}

}

**Output**

1

2

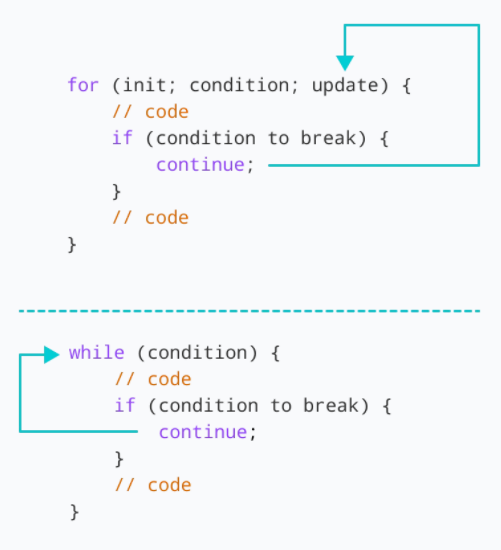
4

5

Here, we have used the continue statement inside the while loop. Similar to the earlier program, when the value of i is **3**, the continue statement is executed.

Hence, **3** is not printed on the screen.

**Working of C# continue Statement**

 Working of continue statement

**continue with Nested Loop**

We use the continue statement with nested as well. For example:

using System;

namespace ContinueNested {

class Program {

static void Main(string[] args) {

int sum = 0;

// outer loop

for(int i = 1; i <= 3; i++) {

// inner loop

for(int j = 1; j <= 3; j++) {

if (j == 2) {

continue;

}

Console.WriteLine("i = " + i + " j = " +j);

}

}

}

}

}

**Output**

i = 1 j = 1

i = 1 j = 3

i = 2 j = 1

i = 2 j = 3

i = 3 j = 1

i = 3 j = 3

In the above example, we have used the continue statement inside the inner for loop. Here, the continue statement is executed when j == 2.

Hence, the value of j = 2 is ignored.

If you want to learn the working of nested loops, visit [C# Nested Loops](https://www.programiz.com/csharp-programming/nested-loops).

**C# continue with foreach Loop**

We can also use the continue statement with foreach loops. For example,

using System;

namespace ContinueForeach {

class Program {

static void Main(string[] args) {

int[] num = { 1, 2, 3, 4, 5 };

foreach(int number in num) {

// skips the iteration

if(number==3) {

continue;

}

Console.WriteLine(number);

}

}

}

}

**Output**

1

2

4

5

In the above example, we have created an array with values: **1**, **2**, **3**, **4**, **5**. Here, we have used the foreach loop to print each element of the array.

However, the loop doesn't print the value **3**. This is because when the number is equal to **3**, the continue statement is executed.

if (number == 3) {

continue;

}

Hence, the print statement for this iteration is skipped.