

# C# Continue Statement with Examples

In **c#**, the **Continue** statement is used to pass control to the next iteration of loops such as for (/tutorial/csharp/csharp-for-loop-with-examples), while (/tutorial/csharp/csharp-while-loop-with-examples), do-while (/tutorial/csharp/csharp-do-while-loop-with-examples), or foreach (/tutorial/csharp/csharp-foreach-loop-with-examples) from the specified position by skipping the remaining code.

In the previous chapter, we learned the break statement in **c#** (/tutorial/csharp/csharp-break-statement-with-examples). The main difference between the break (/tutorial/csharp/csharp-break-statement-with-examples) statement and the **continue** statement is, the break (/tutorial/csharp/csharp-break-statement-with-examples) statement will completely terminate the loop or statement execution. Still, the **continue** statement will pass control to the next iteration of the loop.

## Syntax of C# Continue Statement

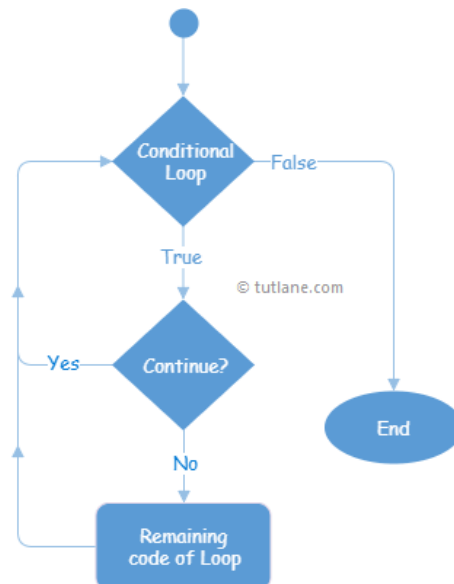
Following is the syntax of defining a continue statement in **c#** programming language.

```
continue;
```

We can use the **continue** statement in our applications whenever we want to skip the code's execution from a particular position and send back the control to the next iteration of the loop based on our requirements.

## C# Continue Statement Flow Chart

Following is the pictorial representation of the **continue** statement process flow in the **c#** programming language.



Now we will see how to use the **continue** statement in for (/tutorial/csharp/csharp-for-loop-with-examples) loop, while (/tutorial/csharp/csharp-while-loop-with-examples) loop, do-while (/tutorial/csharp/csharp-do-while-loop-with-examples) loop, and with switch statement in **c#** programming language with examples.

## C# For Loop with Continue Statement

In **c#**, by using the **continue** keyword, we can skip further code execution and send back control to the next iteration of for (/tutorial/csharp/csharp-for-loop-with-examples) loop statement based on our requirements.

Following is the example of using a **continue** statement with for (/tutorial/csharp/csharp-for-loop-with-examples) loop in **c#** programming language.

```
using System;

namespace Tutlane
{
    class Program
    {
        static void Main(string[] args)
        {
            for (int i = 1; i <= 4; i++)
            {
                if (i == 3)
                    continue;
                Console.WriteLine("i value: {0}", i);
            }
        }
    }
}
```

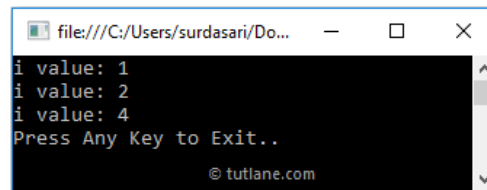
```

        Console.WriteLine("Press Enter Key to Exit..");
        Console.ReadLine();
    }
}
}
}
}

```

If you observe the above code, we used a **continue** statement to pass control back to the next iteration of for (/tutorial/csharp/csharp-for-loop-with-examples) loop whenever the variable **i** value equals **3**.

When we execute the above c# program, we will get the result below.



```

i value: 1
i value: 2
i value: 4
Press Any Key to Exit..
© tutlane.com

```

If you observe the above result, whenever the variable **i** value equals **3**, it skips the further execution of statements and passes the control back to the next iteration of for (/tutorial/csharp/csharp-for-loop-with-examples) loop.

This is how we can use the **continue** statement in for (/tutorial/csharp/csharp-for-loop-with-examples) loop to skip the further execution of statements and send the control back to the further iteration of for loop based on our requirements.

## C# While Loop with Continue Statement

In c#, we can stop executing further statements from the specified position and immediately send the control back to the further iteration of the while (/tutorial/csharp/csharp-while-loop-with-examples) loop.

Following is the example of using the **continue** keyword in a while (/tutorial/csharp/csharp-while-loop-with-examples) loop to pass the control to the next iteration of the loop in the c# programming language.

```

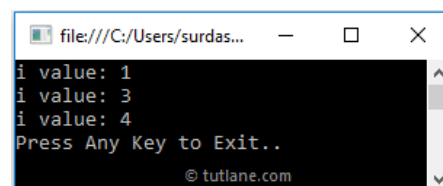
using System;

namespace Tutlane
{
    class Program
    {
        static void Main(string[] args)
        {
            int i = 0;
            while (i < 4)
            {
                i++;
                if (i == 2)
                    continue;
                Console.WriteLine("i value: {0}", i);
            }
            Console.WriteLine("Press Enter Key to Exit..");
            Console.ReadLine();
        }
    }
}

```

If you observe the above example, whenever the variable (**i**) value becomes **2**, we are skipping the further execution of statements and pass the control back to the further iteration of the while (/tutorial/csharp/csharp-while-loop-with-examples) loop using the **continue** statement.

When we execute the above c# program, we will get the result below.



```

i value: 1
i value: 3
i value: 4
Press Any Key to Exit..
© tutlane.com

```

This is how we can use the continue statement with a while (/tutorial/csharp/csharp-while-loop-with-examples) loop to pass the control back to the further iteration of the loop based on our requirements.

## C# Do-While Loop with Continue Statement

Following is the example of using the **continue** keyword in the do...while (/tutorial/csharp/csharp-do-while-loop-with-examples) loop to pass the control to the next iteration of the loop in c# programming language.

```

using System;

namespace Tutlane
{
    class Program
    {
        static void Main(string[] args)
        {
            int i = 1;
            do
            {
                Console.WriteLine("i value: {0}", i);
                i++;
                if (i == 2)
                    continue;
            } while(i < 4);
            Console.WriteLine("Press Enter Key to Exit...");
            Console.ReadLine();
        }
    }
}

```

If you observe the above example, whenever the variable (**i**) value becomes **2**, we are skipping the further execution of statements and pass the control back to the further iteration of the loop using the **continue** statement.

When we execute the above c# program, we will get the result below.

```

i value: 1
i value: 2
i value: 3
Press Enter Key to Exit...

```

This is how we can use the **continue** statement with the do...while (/tutorial/csharp/csharp-do-while-loop-with-examples) loop to pass the control back to the further iteration of the loop based on our requirements.

This is how we can use the **continue** statement in our c# applications to stop the execution of further statements from the specified position and pass control back to the next iteration of the loop based on our requirements.

## CONTACT US

📍 **Address:** No.1-93, Pochamma Colony, Manikonda, Hyderabad, Telangana - 500089

✉ **Email:** support@tutlane.com (mailto:support@tutlane.com)