Increment and Decrement Operators, First Loops

https://csci-1301.github.io/about#authors

June 17, 2021 (08:22:17 AM)

Contents

In this lab you will learn about increment and decrement operators and implement your first while loops.

1 Increment and Decrement Operators

Before writing code, think through the following questions.

Assuming `int i` is initialized and its current value is 5,

1. What does `i++` do to `i`?
2. What does `i--` do to `i`?
3. In your own words, can you explain what is the difference between `++i` and `i++`?

1.1 Practice

Copy and paste this code into a new C# project in your IDE and execute it.

Study the output carefully to make sure you understand the mechanism of the increment and decrement operators.

Compare your answers from previous section to what you observe in the output. Do your answers match with what you observe in the output?

```
using System;

public class IncrementExample
{
    static void Main()
    {
        int a = 0, b = 0;
        Console.WriteLine("Before changing their values:");
        Console.WriteLine($"\ta is {a}\n\tb is {b}\n-----------");
        Console.WriteLine("Incrementing, using postfix and prefix operators:");
```

```
        a++;
        ++b;
        Console.WriteLine($"\ta is {a}\n\tb is {b}\n-----------");
        Console.WriteLine("Decrementing, using postfix and prefix operators:");

        a--;
        --b;
        Console.WriteLine($"\ta is {a}\n\tb is {b}\n-----------");
        Console.WriteLine("When combining decrementing and incrementing operators"
                          + " with other operations,\nit makes a difference whether you
                           ↪ use"
                          + " postfix or prefix operators!");

        int c = a--, d = ++b;
        Console.WriteLine($"\ta is {a} (the decrementing took place as expected)\n"
                          + $"\tb is {b}  (the incrementing took place as expected)\n"
                          + $"\tc is {c}  (c got its value *before* a was decremented)\n"
                          + $"\td is {d}  (d got its value *after* b was incremented)\n"
                          + $"-----------");
    }
}
```

2 First While Loops

1. Write a `while` loop that displays the integers between 1 and 100 on the screen, with a space between them.
2. Write a `while` loop that displays the integers between 100 and −100 on the screen, in decreasing order, with a space between them.
3. Write a `while` loop that displays the ∗ (asterisk) character 100 times on the screen.
4. Modify your previous loop, so that a new line character is displayed on the screen every time 10 asterisks have been displayed on the screen.

That is, your program should display this on the screen (this example has a space after each asterisk for display purposes):

```
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
```

3 Pushing Further (Optional)

Here are additional (fun!) pattern problems. Try generating them using a while loop.

1. Triangle:

```
*
* *
* * *
* * * *
* * * * *
* * * * * *
* * * * * * *
* * * * * * * *
* * * * * * * * *
* * * * * * * * * *
```

2. Triangle of numbers

```
1
222
33333
4444444
555555555
```

3. Upside-down binary triangle

```
1010101
 10101
  101
   1
```