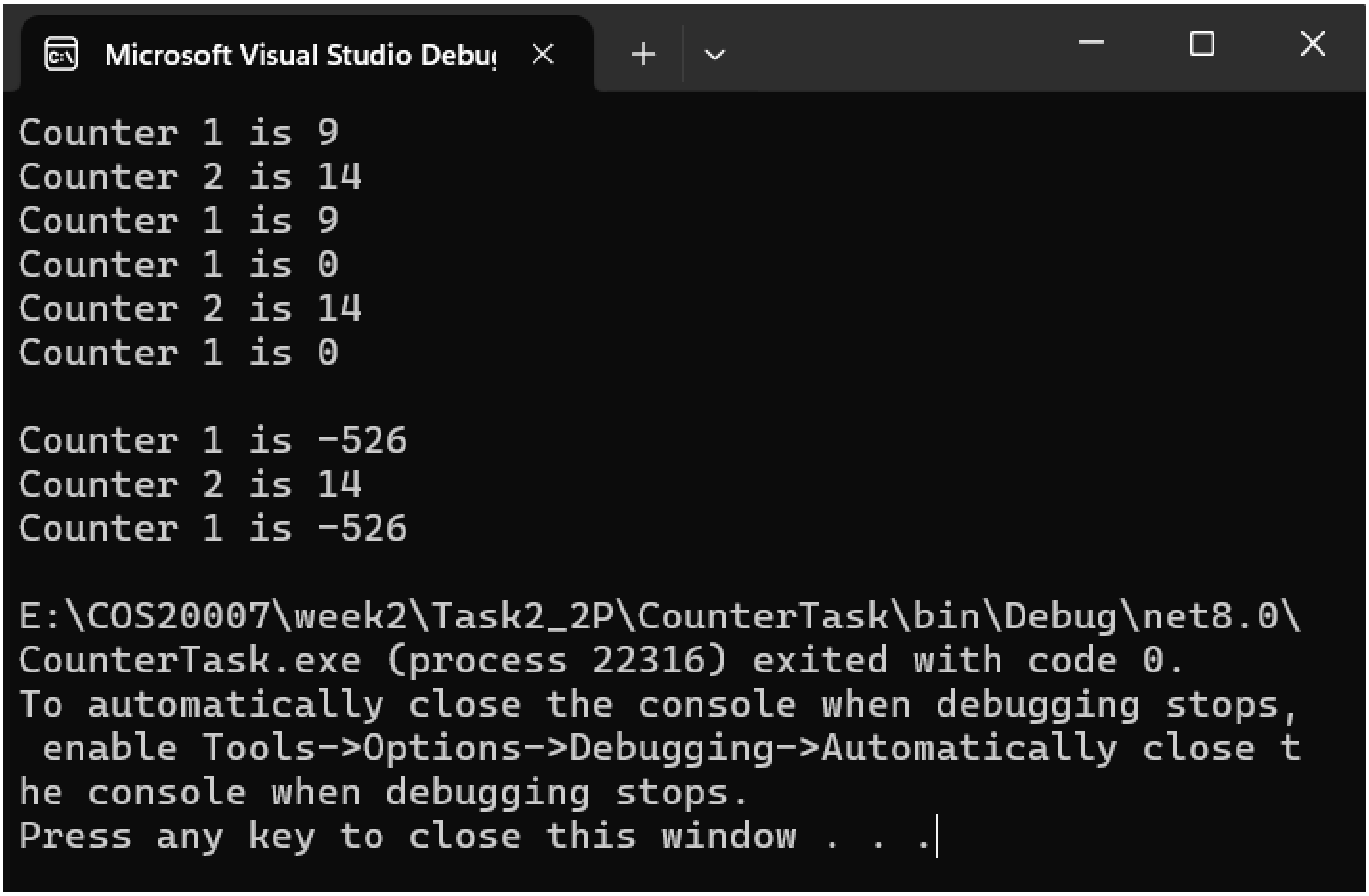
```
public class Counter
{
    private int _count;
    private string _name;

    public Counter(string name)
    {
        _name = name;
        _count = 0;
    }

    public void Increment()
    {
        checked
        {
            _count++;
        }
    }

    public void Reset()
    {
        _count = 0;
    }

    public string Name
    {
        get
        {
            return _name;
        }
        set
        {
            _name = value;
        }
    }

    public int Ticks
    {
        get
        {
            return _count;
        }
    }

    public void ResetByDefault()
    {
        unchecked
        {
            _count = (int)2147483647473;
        }

    }
}
```

```
using System;
namespace CounterTask
{
    internal class Program
    {
        private static void PrintCounters(Counter[] counters)
        {
            foreach (Counter c in counters)
            {
                Console.WriteLine("{0} is {1}", c.Name, c.Ticks);
            }
        }

        static void Main(string[] args)
        {
            Counter[] myCounters= new Counter[3];
            myCounters[0] = new Counter("Counter 1");
            myCounters[1] = new Counter("Counter 2");
            myCounters[2] = myCounters[0];

            for(int i=1; i<10; i++)
            {
                myCounters[0].Increment();
            }

            for(int i=1; i<15; i++)
            {
                myCounters[1].Increment();
            }

            PrintCounters(myCounters);
            myCounters[2].Reset();
            PrintCounters(myCounters);

            Console.WriteLine("");
            myCounters[2].ResetByDefault();
            myCounters[2].Increment();
            PrintCounters(myCounters);
        }
    }
}

```

13. **Tell the Counter to increase the count value. Does the code still run without any bugs/crash? What is the reason behind? You can provide the answers as the comments in the code.**

-The code will still run normally because the "unchecked" method will force the code runner to ignore the overflown value but the value of myCounter[2] will be negative, the reason behind this is because the maximum value that an integer can have in C# is 2,147,483,647 but the value assigned by the ResetByDefault method is bigger(2,147,483,647,473).

Program's output:



```
Counter 1 is 9
Counter 2 is 14
Counter 1 is 9
Counter 1 is 0
Counter 2 is 14
Counter 1 is 0

Counter 1 is -526
Counter 2 is 14
Counter 1 is -526

E:\COS20007\week2\Task2_2P\CounterTask\bin\Debug\net8.0\
CounterTask.exe (process 22316) exited with code 0.
To automatically close the console when debugging stops,
 enable Tools->Options->Debugging->Automatically close t
he console when debugging stops.
Press any key to close this window . . .
```