# Day 14 Assignment

By

VARUN SAI KUMAR CHEGONI

NB Healthcare and Technology

Date: 10 Feb 2022

## Topics

**C# Sealed Class**

**C# Properties**

**C# Break and Continue**

## Content

1. Research and write what is the use of sealed class. WACP to illustrate sealed class.

Answer:

- Sealed class is used to stop a class to be inherited. You cannot derive or extend any class from it.
- Sealed method is implemented so that no other class can overthrow it and implement its own method.
- The main purpose of the sealed class is to withdraw the inheritance attribute from the user so that they can't attain a class from a sealed class. Sealed classes are used best when you have a class with static members.

Code:

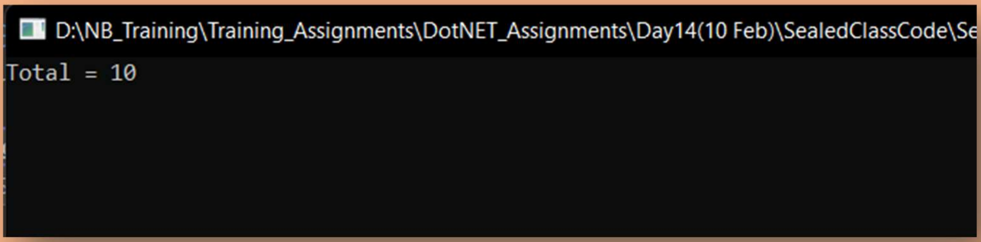
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SealedClassCode
{
    /*****************************************************************************
        * Author : Varun Sai Kumar Chegoni.
        * Purpose : sealed class illustration program
       *****************************************************************************/
    sealed class SealedClass
    {

        // Calling Function
        public int Add(int a, int b)
        {
            return a + b;
        }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            // Creating an object of Sealed Class
            SealedClass slc = new SealedClass();

            // Performing Addition operation
            int total = slc.Add(6, 4);
            Console.WriteLine("Total = " + total.ToString());
            Console.ReadLine();
        }
    }
}
```

Output :



```
D:\NB_Training\Training_Assignments\DotNET_Assignments\Day14(10 Feb)\SealedClassCode\Se
Total = 10
```

2. Research and write what is the difference between normal properties and auto-implemented properties.
WACP to illustrate normal properties.
WACP to illustrate auto-implemented properties.

Answer:

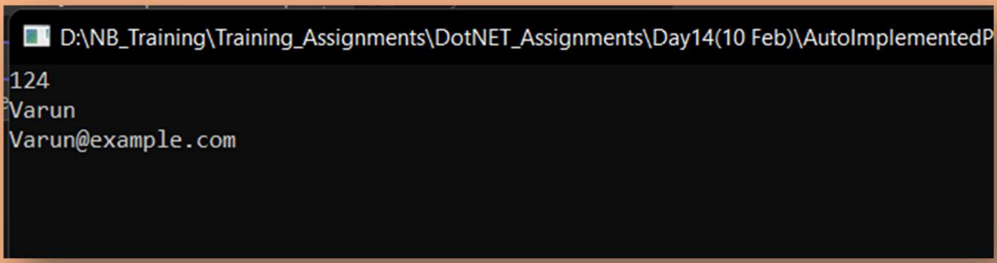
| Normal Properties | Auto-Implemented Properties |
|---|---|
| 1. Normal properties refers to the private variables | 1. Auto implemented properties will not refer to any private variables. |
| 2. In normal properties we can either take get or set or both get and set as well. | 2. In auto implemented properties we must take either get or both set and get. |

Code to illustrate auto-implemented properties:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AutoImplementedProperty
{
    class Student
    {
        // Auto-implimented Properties
        public int ID { get; set; }
        public string Name { get; set; }
        public string Email { get; set; }
    }
    internal class Program
    {
        static void Main(string[] args)
        {
            Student student = new Student();
            // Setting properties
            student.ID    = 124;
            student.Name  = "Varun";
            student.Email = "Varun@example.com";
            // Getting properties
            Console.WriteLine(student.ID);
            Console.WriteLine(student.Name);
            Console.WriteLine(student.Email);
            Console.ReadLine();
        }
    }
}
```

Output :



```
D:\NB_Training\Training_Assignments\DotNET_Assignments\Day14(10 Feb)\AutoImplementedP
124
Varun
Varun@example.com
```

3. WACP to check if the number is prime or not using logic discussed in the class.
HINT : use break;

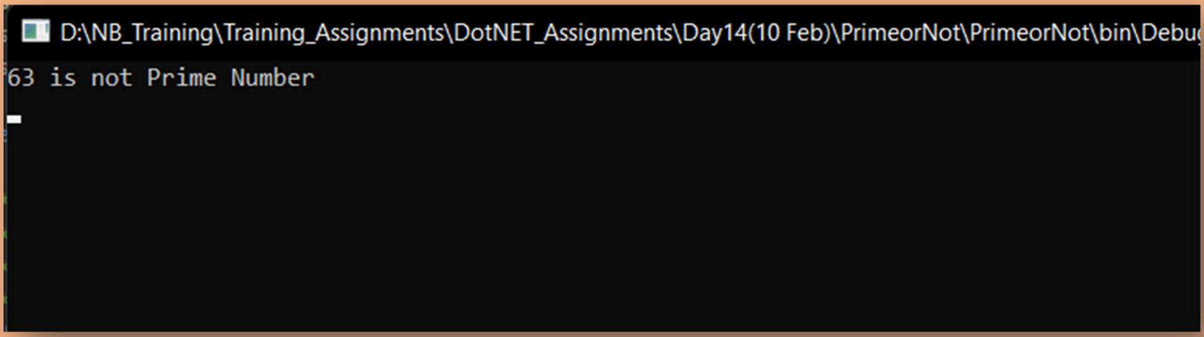
Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace PrimeorNot
{
    /***************************************************************************
        * Author : Varun Sai Kumar Chegoni.
        * Purpose : number is prime or not using logic discussed in the class use
break;
       ***************************************************************************/
    internal class Program
    {
        static void Main(string[] args)
        {
            int i, n = 63;
            for (i=2; i<n; i++)
            {
                if (n % i == 0)
                    break;
            }
            if (i==n)
                Console.WriteLine("63 is a Prime Number");
            else
                Console.WriteLine("63 is not Prime Number");
            Console.ReadLine();

        }
    }
}
```

Output :



```
D:\NB_Training\Training_Assignments\DotNET_Assignments\Day14(10 Feb)\PrimeorNot\PrimeorNot\bin\Debu
63 is not Prime Number
```

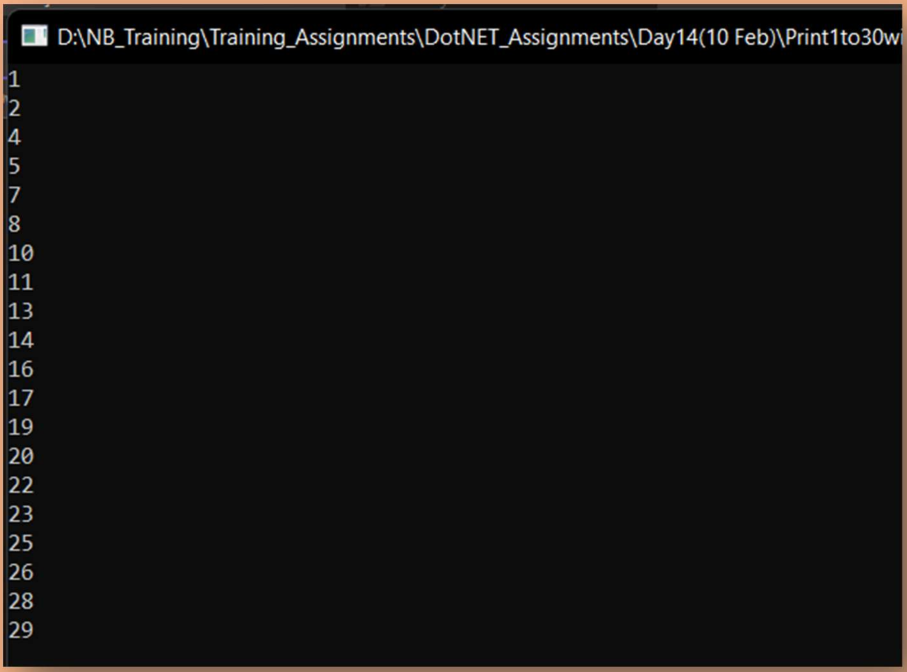
4. print numbers from 1 to 30 and skip the numbers divisible by 3
HINT: use continue;

Code:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Print1to30without3X
{
    /***********************************************************************************
        * Author : Varun Sai Kumar Chegoni.
        * Purpose : print numbers from 1 to 30 and skip the numbers divisible by 3
use continue;
       ********************************************************************************/
    internal class Program
    {
        static void Main(string[] args)
        {
            int n;
            for (int i = 1; i<=30; i++)
            {
                if (i % 3 == 0)
                    continue;
                Console.WriteLine(i);
            }
            Console.ReadLine();
        }
    }
}
```

Output :



```
D:\NB_Training\Training_Assignments\DotNET_Assignments\Day14(10 Feb)\Print1to30wi
1
2
4
5
7
8
10
11
13
14
16
17
19
20
22
23
25
26
28
29
```

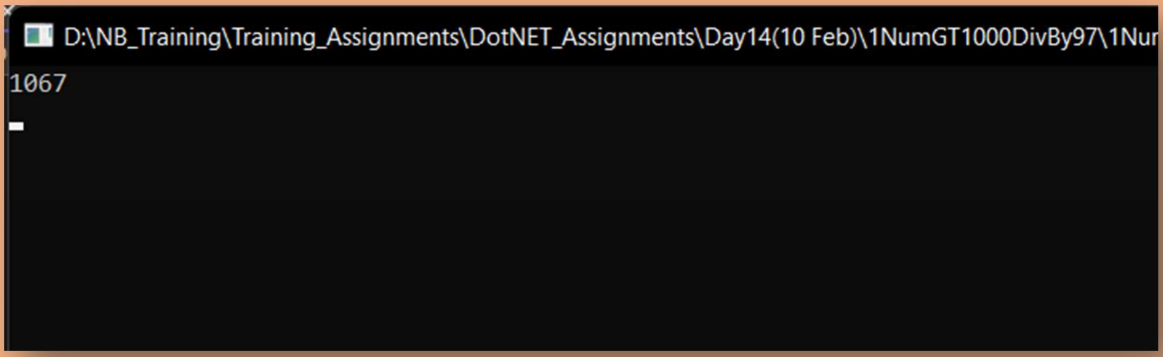
5. Find the first number after 1000 which is divisible by 97.
HINT: use for loop and break

Code:

```csharp
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace _1NumGT1000DivBy97
{
    /*****************************************************************************
        * Author : Varun Sai Kumar Chegoni.
        * Purpose : first number after 1000 which is divisible by 97
       *****************************************************************************/
    internal class Program
    {
        static void Main(string[] args)
        {
            int n = 97;
            for (int i = 1000; i<=1097; i++)
            {
                if (i%n==0)
                {
                    Console.WriteLine(i);
                    break;
                }
            }
            Console.ReadLine();
        }
    }
}
```

Output :



```
D:\NB_Training\Training_Assignments\DotNET_Assignments\Day14(10 Feb)\1NumGT1000DivBy97\1Nu
1067
```