Day 14 Assignment

by

Vinay Kudali

11-02-2022

nations benefits

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

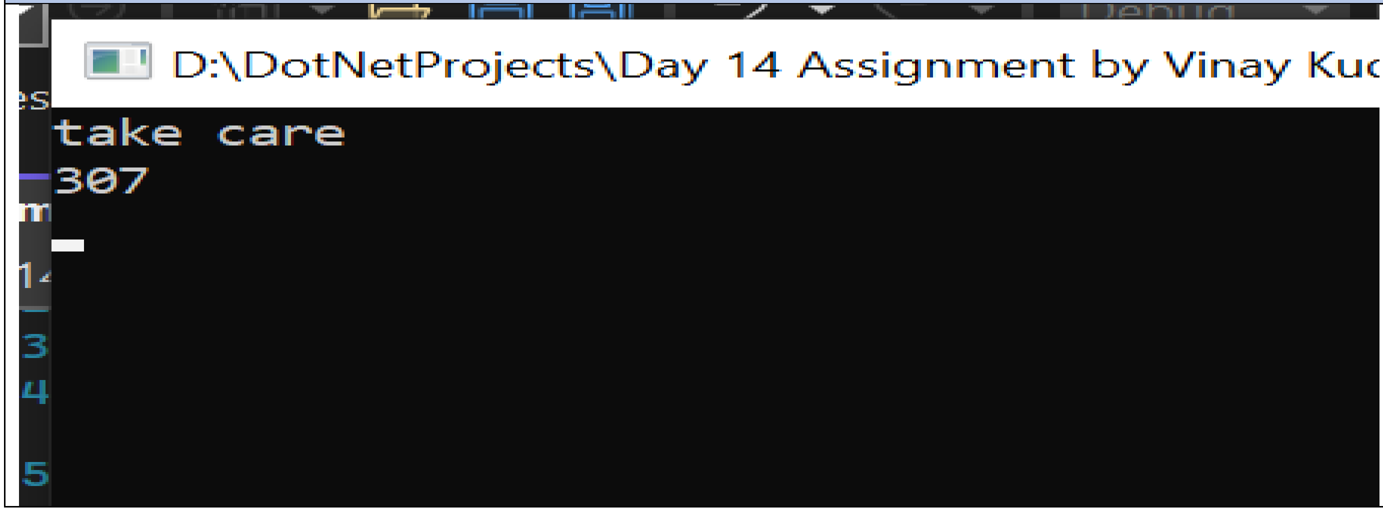
- A sealed class can have variables as well as normal class.
- But, the thing difference between normal class and sealed class is cannot be used as Super class or parent class for another class
- Inheritance is not possible in Sealed class.

WACP to illustrate sealed class.

Code:

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

namespace Day14Project1SealedClass
{
    //Author: Vinay Kudali
    //Purpose:Creating sealed class
    sealed class Hotel
    {
        public int roomNo = 307;

        public string Message()
        {
            return "take care";
        }
    }


    internal class Program
    {
        static void Main(string[] args)
        {
            Hotel h = new Hotel();
            Console.WriteLine(h.Message());
            Console.WriteLine(h.roomNo);
            Console.ReadLine();
        }
    }
}
```

Output:



```
D:\DotNetProjects\Day 14 Assignment by Vinay Kuc
take care
307
```

2. Research and write what is the differencebetween normal properties and auto-implementedproperties.

Normal Properties	Auto-Implemented Properties
• Write only- When property contains only "**Set**". • Read Only- When property contains Only "**get**". • Generally, Normal Properties are used to Access Private Variables.	• Auto implemented properties must have "**get**" Accessors. • "**Set**" is optional, "**get**" is Mandatory. • Auto-Implemented properties will not do point any other variables.

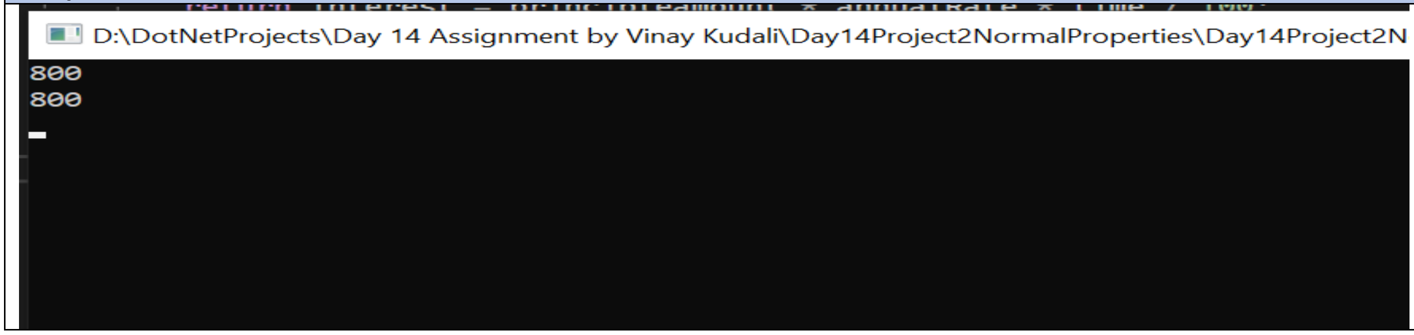
WACP to illustrate normal properties
WACP to illustrate auto-implemented properties

Code:

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

namespace Day14Project2NormalProperties
{
  class SimpleInterest
```

```
{
    //Author: Vinay Kudali
    //Purpose Creating Class By using Normal Properties and Auto-Implemented Properties
    private int principleamount;
    private int annualRate;
    private int time;
    private int Interest;
    //Normal Properties
    public int Principleamount
    {
        set
        {
            principleamount = value;
        }
    }
    public int AnnualRate
    {
        set
        {
            AnnualRate = value;
        }
    }
    public int Time
    {
        set
        {
            time = value;
        }
    }
    public int interest
    {
        get
        {
            return Interest = principleamount * annualRate * time / 100;
        }
    }
    //Auto-Implemented Properties
    public int AutoImplementedInterest
    {
        get
        {
            return Interest = principleamount * annualRate * time / 100;
        }

    }
    internal class Program
    {
        static void Main(string[] args)
        {
            SimpleInterest s = new SimpleInterest();
            s.principleamount = 10000;
            s.annualRate = 4;
            s.time = 2;
            Console.WriteLine(s.interest);
            Console.WriteLine(s.AutoImplementedInterest);
            Console.ReadLine();
        }
    }
}
}
```

Output:



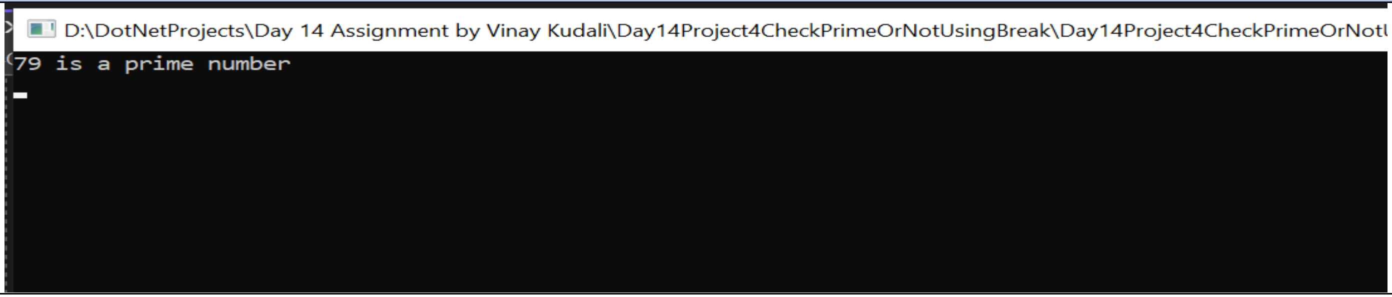
```
D:\DotNetProjects\Day 14 Assignment by Vinay Kudali\Day14Project2NormalProperties\Day14Project2N
800
800
```

4. WACP to check if the number is prime or not using logic discussed in the classHINT: use break;

Code:

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

namespace Day14Project4CheckPrimeOrNotUsingBreak
{
    class Program
    {

        //Author: Vinay Kudali
        //Purpose: Findout a number is prime or not by using break

        static void Main(string[] args)
        {
            int i, n = 79;
            for (i = 2; i < n; i++)
            {
                if (n % i == 0)
                    break;
            }
            if (i == n)
                Console.WriteLine("{0} is a prime number", n);
            else
                Console.WriteLine("{0} is not a prime number", n);
            Console.ReadLine();

        }
    }
}
```

Output:



```
D:\DotNetProjects\Day 14 Assignment by Vinay Kudali\Day14Project4CheckPrimeOrNotUsingBreak\Day14Project4CheckPrimeOrNotU
79 is a prime number
```

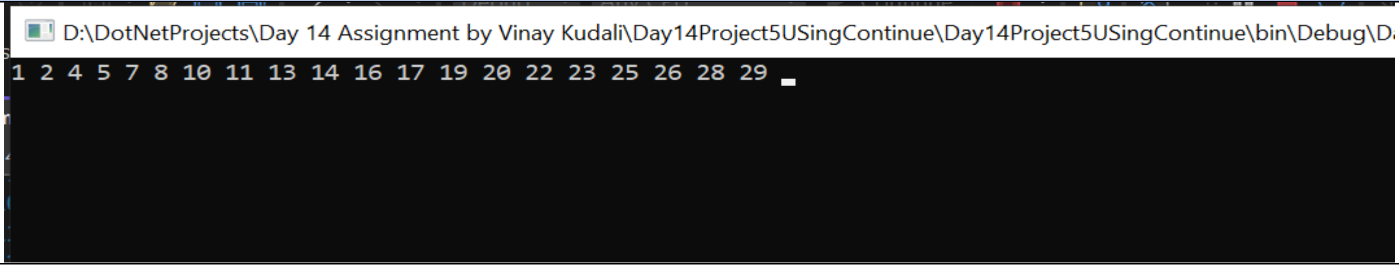
5. print numbers from 1 to 30 and skip the numbers divisible by 3 HINT: use continue;

Code:

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

```
namespace Day14project5UsingContinue
{
   class Program
   {

      //Author:Vinay Kudali
      //Purpose: Printing the values which are not divisible by 3 using continue


      static void Main(string[] args)
      {
         int n = 30;
         for (int i = 1; i <= n; i++)
         {
            if (i % 3 == 0)
               continue;
            Console.Write(i + " ");

         }
         Console.ReadLine();
      }
   }
}
```

Output:



```
D:\DotNetProjects\Day 14 Assignment by Vinay Kudali\Day14Project5USingContinue\Day14Project5USingContinue\bin\Debug\D
1 2 4 5 7 8 10 11 13 14 16 17 19 20 22 23 25 26 28 29 _
```

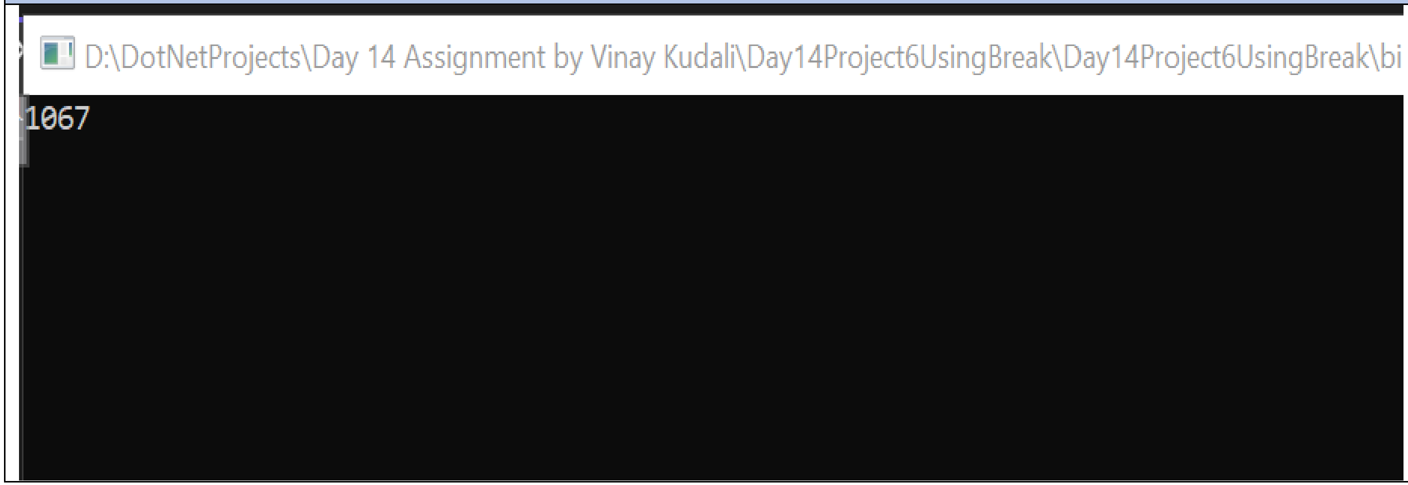
6. Find the first number after 1000 which isdivisible by 97.HINT: use for loop and break

Code:

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

```
namespace Day14project6UsingBreak
{
    class Program
    {

        //Author:Vinay Kudali
        //Purpose: Find the first number later 1000 which is divisible by 97 using for loop and break

        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:\DotNetProjects\Day 14 Assignment by Vinay Kudali\Day14Project6UsingBreak\Day14Project6UsingBreak\bi

```
1067
```