Day 14 Assignment
By
M Mary Margarette
On 10-02-2022

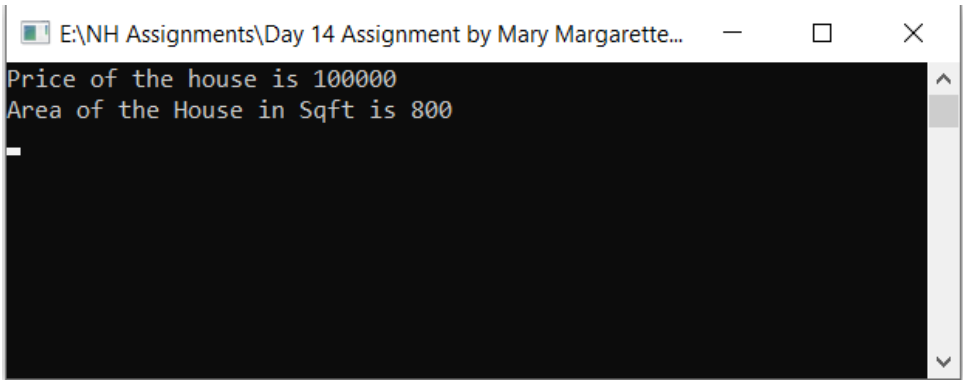
Research and write what is the use of sealed class. WACP to illustrate sealed class.
❖ Sealed class is same as normal class but it cannot be used as base class / parent class.
❖ If we try to inherit a sealed class with another class it fails due to it is sealed and can't be used as a base class.
❖ It can be accessed only through the class which is sealed. i.e., itself only.
❖ Sealed class is used for security purposes.
❖ No class can be derived from a sealed class.
❖ It restricts the users from inheriting class.

Code:

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

namespace Day_14_Project_4
{
   //Author: Mary Margaret
   //using Sealed class
   internal class Program
   {
      sealed class House
      {
         public static int Sqft = 800;
         public int GetPrice()
         {
```

```
            return 100000;
        }
    }

    static void Main(string[] args)
    {
        //object creation
        House h = new House();
        Console.WriteLine( "Price of the house is {0}", h.GetPrice() );
        Console.WriteLine("Area of the House in Sqft is {0}", House.Sqft);

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

Output:



Research and write what is the difference between normal properties and auto implemented properties.

→ Auto-implemented properties enable you to quickly specify a property of a class without writing code for Get and Set the property.

→ The compiler creates a private anonymous field that can only be accessed through the property's accessors.

❖ In Normal Properties the user should initialize getters and setters and the user should assign values.

- ❖ Normal Properties are properties without having any backing fields.

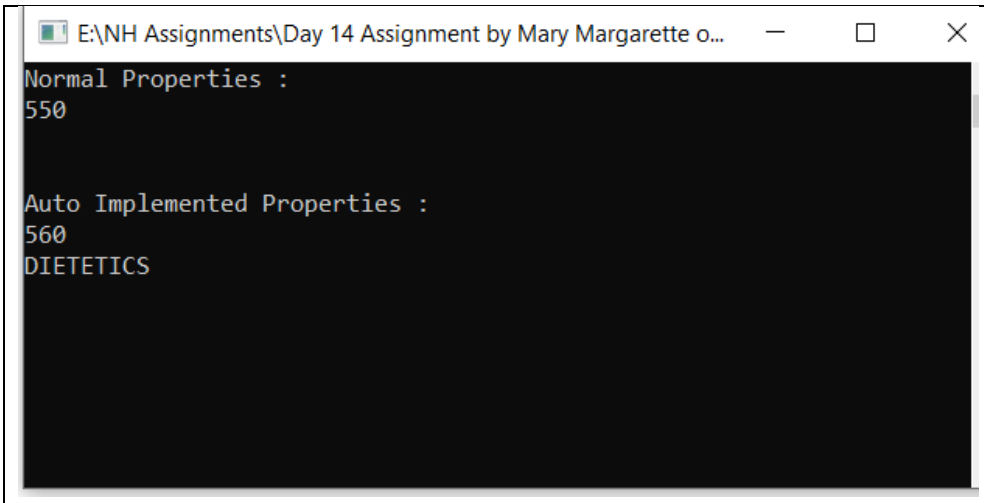
WACP to illustrate normal Properties and Auto implemented Properties.

Code:

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

//Author : Mary Margaret
//Normal Properties
namespace Day_14_Project_5
{
   /// <summary>
   /// Normal Properties
   /// </summary>
   class Normal
   {
      private int price;
      public int Price
      {
         get
         {
            return price;
         }
         set
         {
            price = value;
         }
      }
      public int Book
      {
         set
         {
            Book = value;
         }
      }
   }

   /// <summary>
   ///Auto Implemented Properties
   /// </summary>
   class Auto
   {
```

```
    public int Price { get; set; }
    public string Book
    {
      get
      {
        return "DIETETICS";

      }
    }
  }

  internal class Program
  {
    static void Main(string[] args)
    {
      //Object Creation for Normal Properties
      Console.WriteLine("Normal Properties :");
      Normal n = new Normal();
      n.Price = 550;
      Console.WriteLine(n.Price);
      Console.WriteLine("\n");

      //Object Creation for Auto Implemented Properties
      Console.WriteLine("Auto Implemented Properties :");
      Auto a = new Auto();
      a.Price = 560;
      Console.WriteLine(a.Price);
      Console.WriteLine(a.Book);
      Console.ReadLine();
    }
  }
}
```

Output:

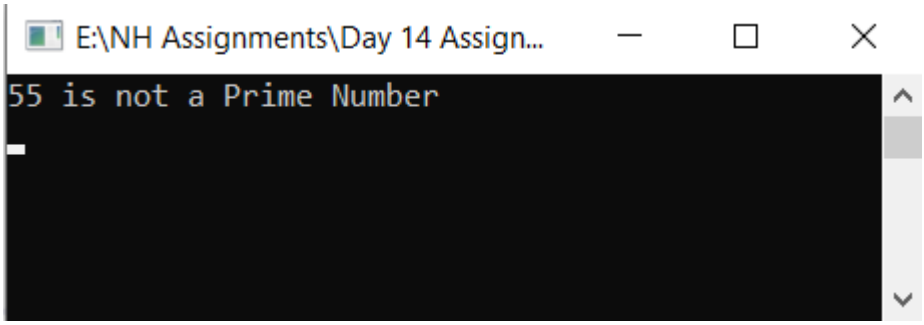


WACP to check if the number is prime or not using logic discussed in the class HINT: use break;
Code:

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

namespace Day_14_Project_2
{
   //Author: Mary Margaret
   // To check if the number is prime or not using break;
   internal class Program
   {
      static void Main(string[] args)
      {
         int n = 55;
         int i;
         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:



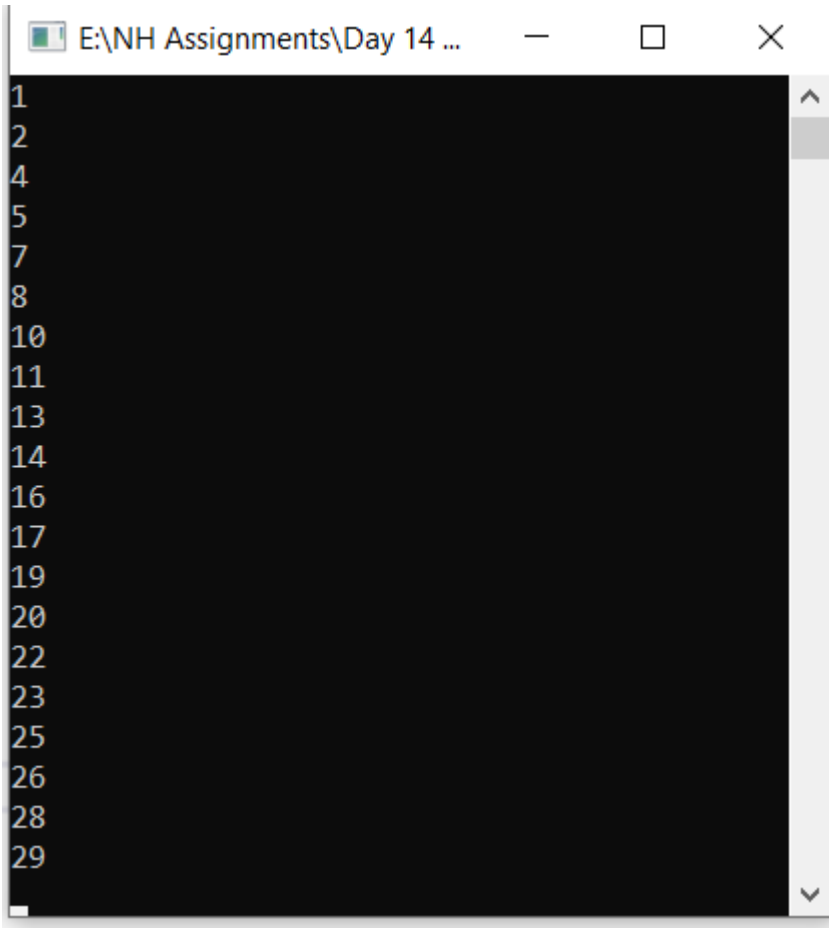
print numbers from 1 to 30 and skip the numbers divisible by 3HINT: use continue;

Code:

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

namespace Day_14_Project_1
{
  //Author: Mary Margaret
  //printing 1 to 30 num using continue; and skipping div 3 numbers.
  internal class Program
  {

    static void Main(string[] args)
    {
      int n = 1;
```

```
        for(int i = 1; i <= 30; i++)
        {
           if (i % 3 == 0)
              continue;
           Console.WriteLine(i);
        }
        Console.ReadLine();
     }
  }
}
```

Output:

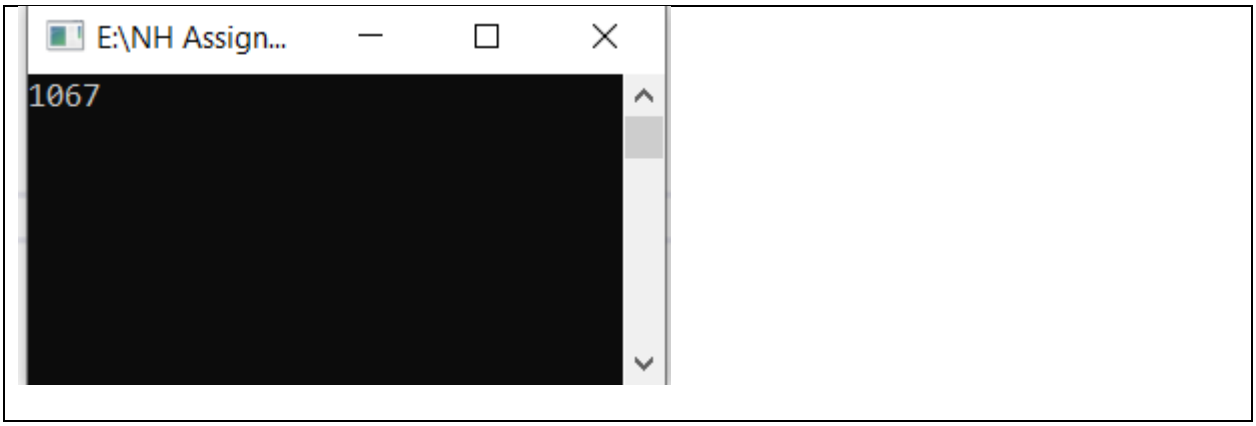


Find the first number after 1000 which is divisible by 97.
HINT: use for loop and break

Code:

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

namespace Day_14_Project_3
{
   //Author : Mary Margaret
   //Num div by 97 from 1000 to 1097
   internal class Program
   {
      class Break
      {
         int n = 97;
         int i;
         public void Div()
         {
            for ( i = 1000; i <= 1097; i++)
            {
               if (i % n == 0)
                  break;

            }
            Console.WriteLine(i);

         }

   }
      static void Main(string[] args)
      {
         Break b = new Break();
         b.Div();

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

Output:


E:\NH Assign...
1067