

DAY14 ASSIGNMENT
BY
PAVAN KUMAR (10-02-2022)

Q1). Research and write what is the use of sealed class. WACP to illustrate sealed class.

SEALED CLASS:

A sealed class, in C#, is a class that cannot be inherited by any class but can be instantiated. The keyword used is Sealed, the keyword tells the compiler that the class is sealed, and therefore, cannot be extended.

- It cannot be used as a Base class for another class.
- Sealed classes are used best when you have a class with static members.

CODE:

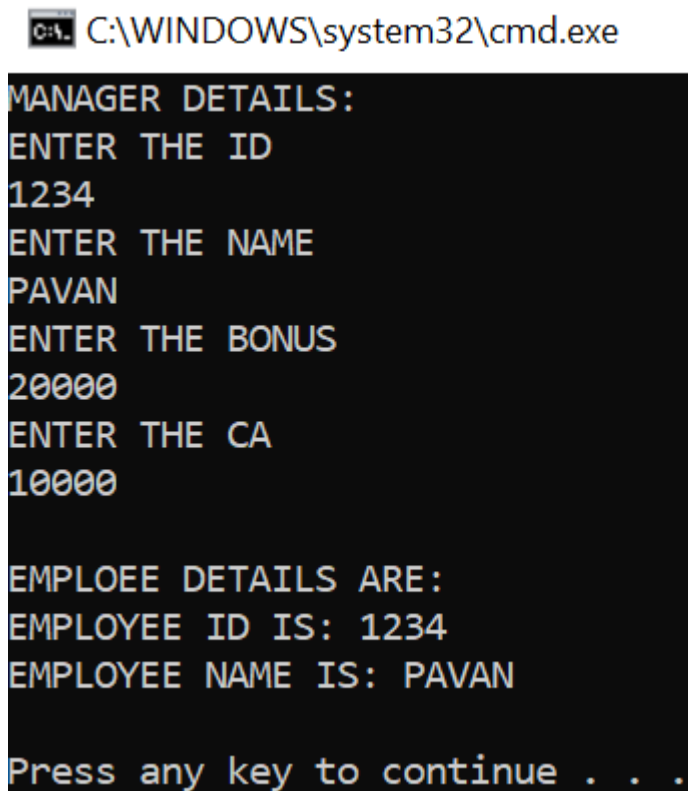
```
using System;
using System.Collections.Generic;
namespace SealedDemo
{
    /// <summary>
    /// DONE BY: PAVAN
    /// PURPOSE: CREATING A SEALED CLASS IN WHICH MANAGER CAN ACCESS EMPLOYEE
    CLASS//
    /// </summary>
    public class Employee
    {
        public int id;
        public string name;
        public virtual void GetEmployeeData()
        {
            Console.WriteLine("EMPLOYEE DETAILS:");
            Console.WriteLine("ENTER THE ID");
            id = Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("ENTER THE NAME");
            name = Console.ReadLine();
        }
        public virtual void DisplayEmployeeData()
        {
            Console.WriteLine("\nEMPLOYEE DETAILS ARE:");
            Console.WriteLine("EMPLOYEE ID IS: " + id);
            Console.WriteLine("EMPLOYEE NAME IS: " + name);
        }
    }
    public sealed class Manager: Employee
    {
        public override void GetEmployeeData()
        {
```

```

        Console.WriteLine(" ENTER MANAGER DETAILS:");
        Console.WriteLine("ENTER THE ID");
        id = Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("ENTER THE NAME");
        name = Console.ReadLine();
    }
}
internal class Program
{
    static void Main(string[] args)
    {
        Manager m = new Manager();
        m.GetEmployeeData();
        m.DisplayEmployeeData();
        Console.ReadLine();
    }
}

```

OUTPUT:



```

C:\WINDOWS\system32\cmd.exe

MANAGER DETAILS:
ENTER THE ID
1234
ENTER THE NAME
PAVAN
ENTER THE BONUS
20000
ENTER THE CA
10000

EMPLOYEE DETAILS ARE:
EMPLOYEE ID IS: 1234
EMPLOYEE NAME IS: PAVAN

Press any key to continue . . .

```

PROJECT: 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;

```

```

namespace Day14Project2
{
    /// <summary>
    /// DONE BY: PAVAN
    /// PURPOSE: USING BREAK CHECK THE GIVEN NUMBER IS PRIME OR NOT PRIME//
    /// </summary>
    internal class Program
    {
        static void Main(string[] args)
        {
            int n = 2, i;
            for (i = 2; i < n; i++)
            {
                if (n % i == 0)
                    break;
            }
            if (i == n)

                Console.WriteLine(" ITS A PRIME NUMBER:");
            else
                Console.WriteLine("NOT A PRIME NUMBER:");
            Console.ReadLine();
        }
    }
}

```

OUTPUT:

C:\WINDOWS\system32\cmd.exe

ITS A PRIME NUMBER:

PROJECT: 5

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

CODE:

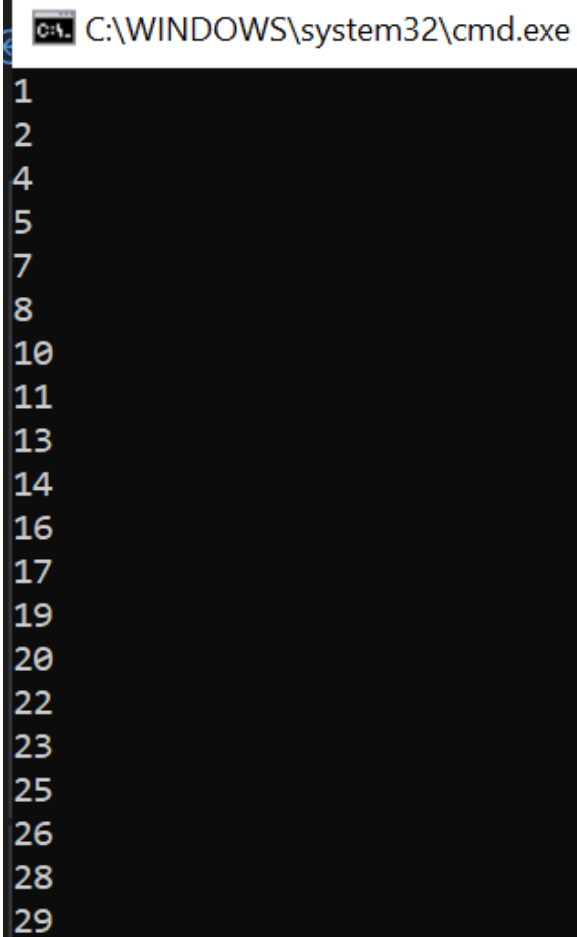
```

using System;
using System.Collections.Generic;
namespace Day14Project3
{
    internal class Program
    {
        /// <summary>
        /// DONE BY: PAVAN
        /// PURPOSE: USING CONTINUE KEY print numbers from 1 to 30 and skip the numbers divisible by 3
        /// </summary>

```

```
/// <param name="args"></param>
static void Main(string[] args)
{
    for (int i = 1; i <= 30; i++)
    {
        if (i % 3 == 0)
            continue;
        Console.WriteLine(i);
    }
    Console.ReadLine();
}
}
```

OUTPUT:



C:\WINDOWS\system32\cmd.exe

1
2
4
5
7
8
10
11
13
14
16
17
19
20
22
23
25
26
28
29

PROJECT: 6

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

CODE:

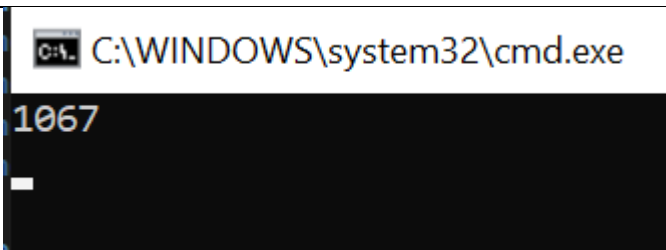
```
using System;
using System.Collections.Generic;
```

```

namespace Day14Project4
{
    internal class Program
    {
        /// <summary>
        /// DONE BY: PAVAN
        /// PURPOSE: Find the first number after 1000 which is divisible by 97. HINT: use for loop and break//
        /// </summary>
        /// <param name="args"></param>
        static void Main(string[] args)
        {
            for (int i = 1000; i<=1097; i++)
            {
                if (i % 97 == 0)
                {
                    Console.WriteLine(i);
                    break;
                }
            }
            Console.ReadLine();
        }
    }
}

```

OUTPUT:



PROJECT: 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.

DIFFERENCE BETWEEN NORMAL PROPERTIES AND AUTO-IMPLEMENTING PROPERTIES:

- ❖ In normal property both **{get, set}** accessor is present.
- ❖ Auto-Implementing properties must have **{get}** accessor.
- ❖ **Read-Only Properties:** When property contains only get method.
- ❖ **Write Only Properties:** When property contains only set method.

CODE:

```

using System;
using System.Collections.Generic;
namespace Day14Project5
{

```

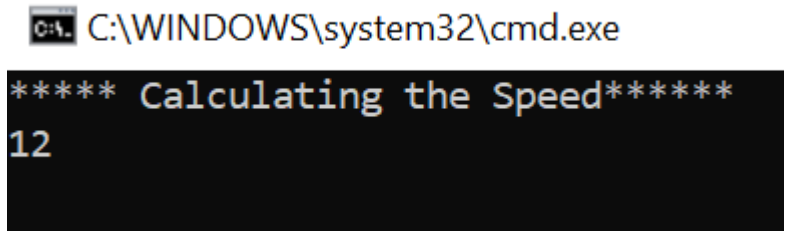
```

class SpeedCalculation
{
    /// <summary>
    /// DONE BY: PAVAN
    /// PURPOSE: WRITE A C# PROGRAM ON AUTO-IMPLEMENTED PROPERTY.
    /// </summary>
    /// <param name="args"></param>
    private int time;
    private int distance;

    public int Time
    {set {time = value;}}
    public int Distance
    {set {distance = value;}}
    public int Speed
    {get {return distance / time;}}
    public int Velocity
    {get {return distance / time;} set { }}
}
internal class program
{
    static void Main(string[] args)
    {
        SpeedCalculation sc = new SpeedCalculation();
        sc.Time = 57;
        sc.Distance = 684;
        Console.WriteLine("***** Calculating the Speed*****");
        Console.WriteLine($"{sc.Speed}");
        Console.ReadLine();
    }
}

```

OUTPUT:



C:\WINDOWS\system32\cmd.exe

```

***** Calculating the Speed*****
12

```