Tribhuvan University

Faculty of Humanities and Social Sciences

DOT NET LABSHEETS

A PROJECT REPORT

Submitted to

Department of Computer Application

Shahid Smarak College

In partial fulfillment of the requirements for the Bachelors in Computer Application

Submitted by: -

Amir Maharjan

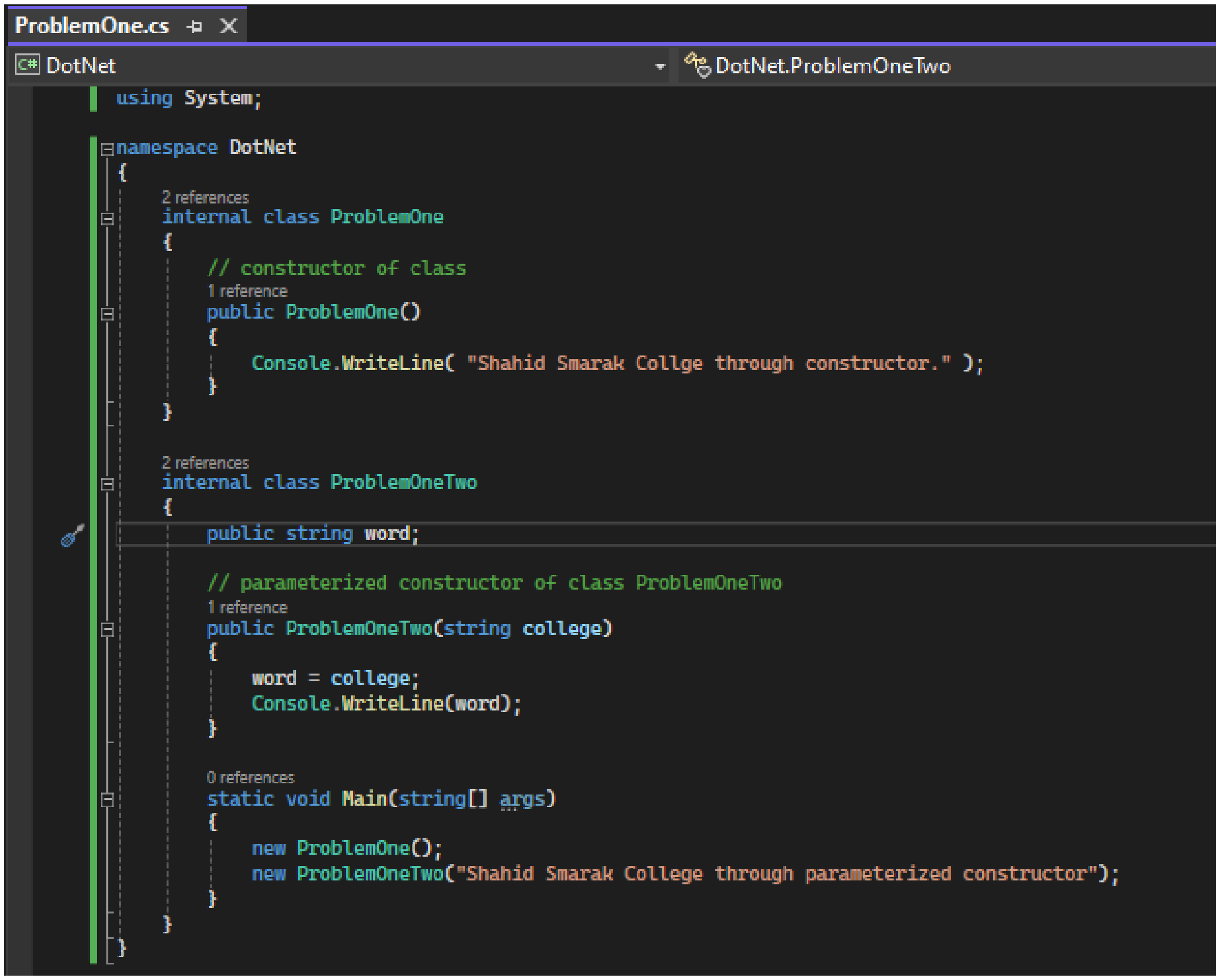
Internal supervisor **External Supervisor**

Rajesh Shahi Thakuri

Problem – 1

Write a program to implement the concept of default constructor, parameterized constructor

Code

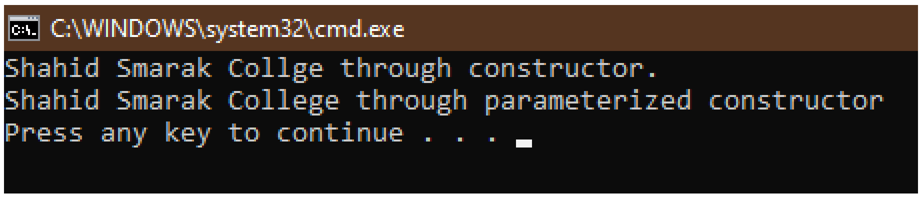


ProblemOne.cs

DotNet | DotNet.ProblemOneTwo

```
using System;

namespace DotNet
{
    2 references
    internal class ProblemOne
    {
        // constructor of class
        1 reference
        public ProblemOne()
        {
            Console.WriteLine( "Shahid Smarak Collge through constructor." );
        }
    }

    2 references
    internal class ProblemOneTwo
    {
        public string word;

        // parameterized constructor of class ProblemOneTwo
        1 reference
        public ProblemOneTwo(string college)
        {
            word = college;
            Console.WriteLine(word);
        }

        0 references
        static void Main(string[] args)
        {
            new ProblemOne();
            new ProblemOneTwo("Shahid Smarak College through parameterized constructor");
        }
    }
}
```

Result



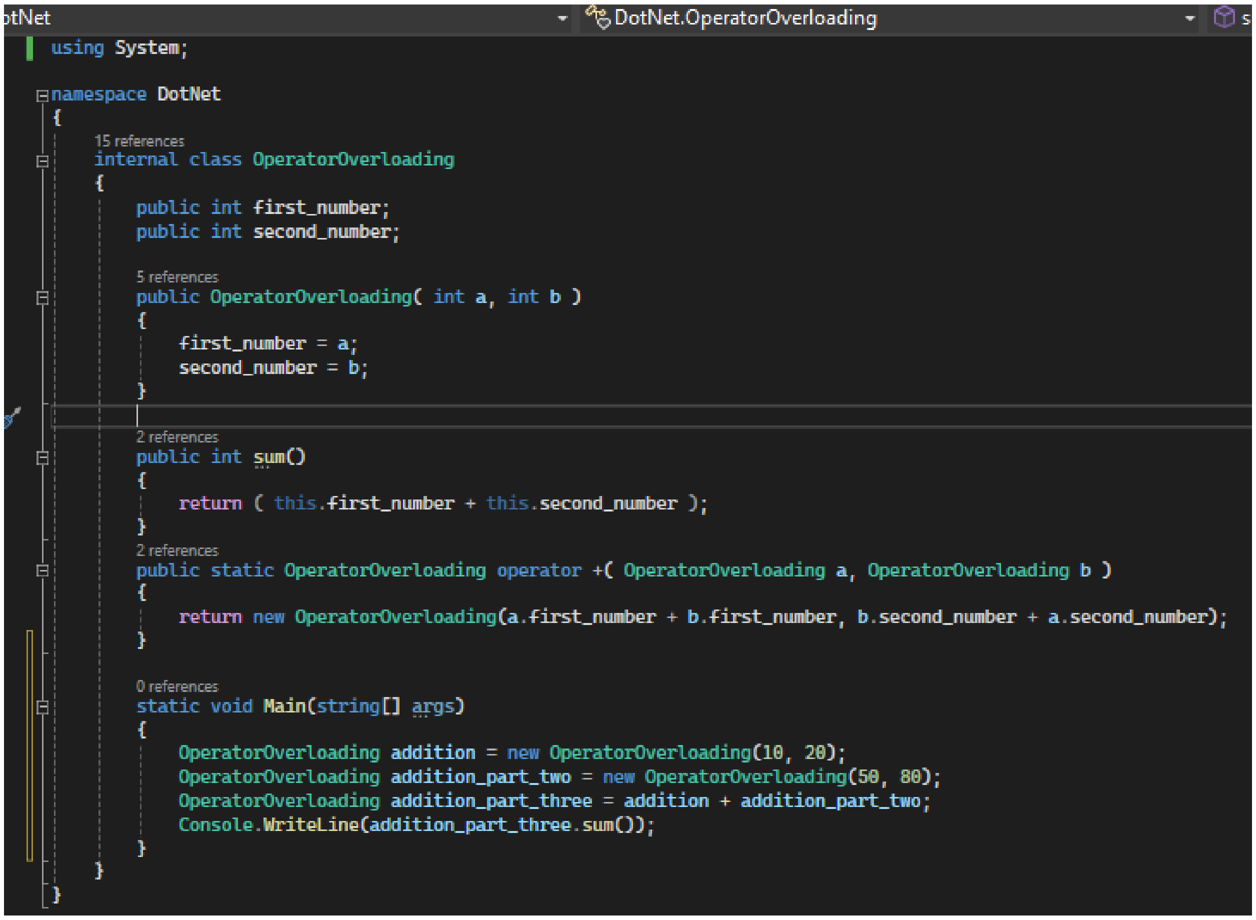
C:\WINDOWS\system32\cmd.exe

```
Shahid Smarak Collge through constructor.
Shahid Smarak College through parameterized constructor
Press any key to continue . . .
```

Problem – 2

Write a program to implement the concept to operator (+) overloading (binary)

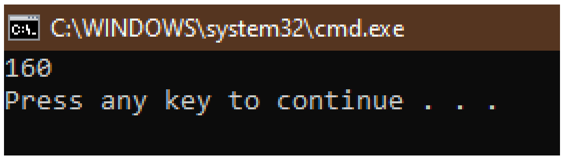
Code



otNet DotNet.OperatorOverloading

```
using System;

namespace DotNet
{
    15 references
    internal class OperatorOverloading
    {
        public int first_number;
        public int second_number;

        5 references
        public OperatorOverloading( int a, int b )
        {
            first_number = a;
            second_number = b;
        }

        2 references
        public int sum()
        {
            return ( this.first_number + this.second_number );
        }
        2 references
        public static OperatorOverloading operator +( OperatorOverloading a, OperatorOverloading b )
        {
            return new OperatorOverloading(a.first_number + b.first_number, b.second_number + a.second_number);
        }

        0 references
        static void Main(string[] args)
        {
            OperatorOverloading addition = new OperatorOverloading(10, 20);
            OperatorOverloading addition_part_two = new OperatorOverloading(50, 80);
            OperatorOverloading addition_part_three = addition + addition_part_two;
            Console.WriteLine(addition_part_three.sum());
        }
    }
}
```

Result



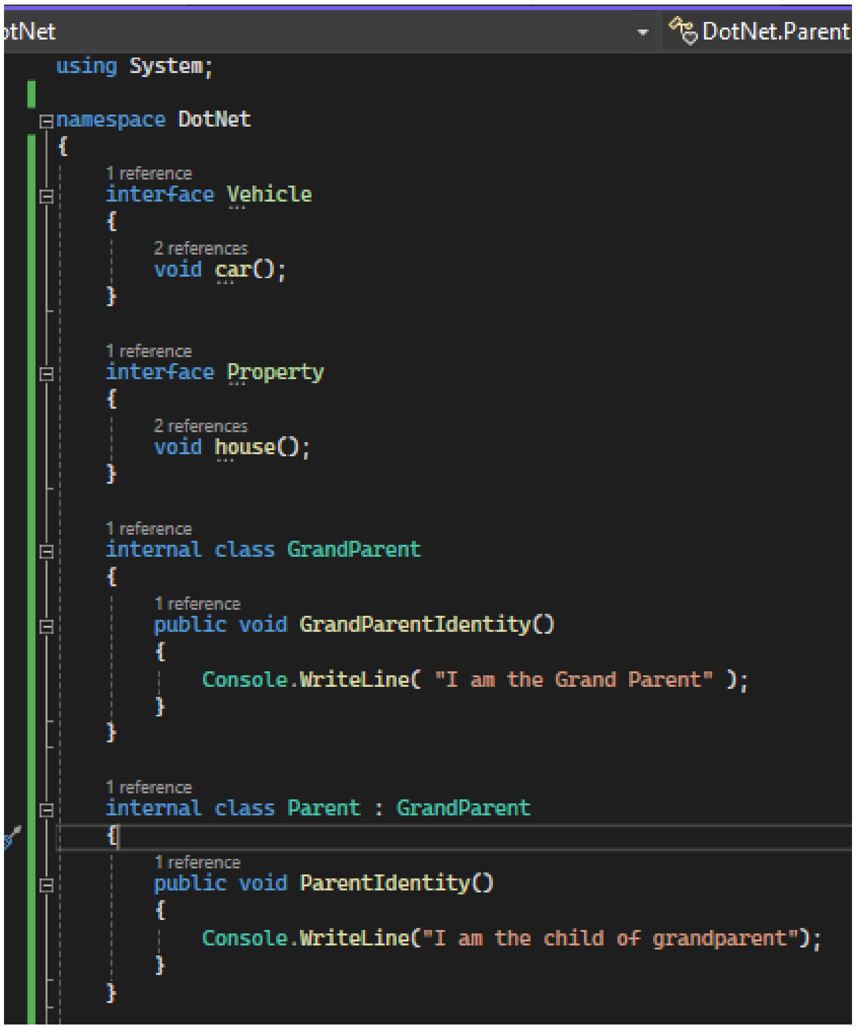
C:\WINDOWS\system32\cmd.exe

```
160
Press any key to continue . . .
```

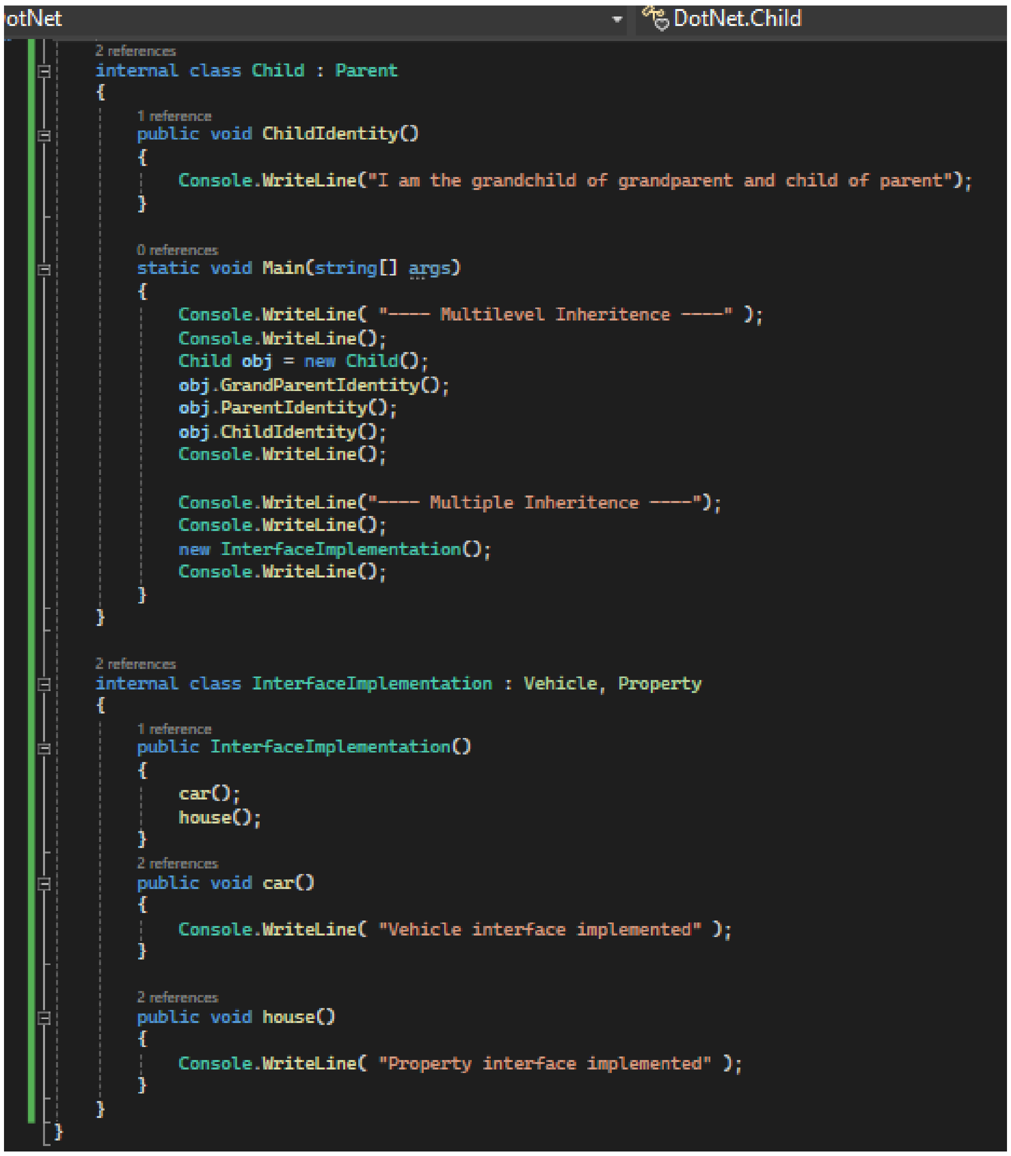
Problem – 3

Write a program to show the concept of multilevel inheritance and multiple inheritance in c#

Code

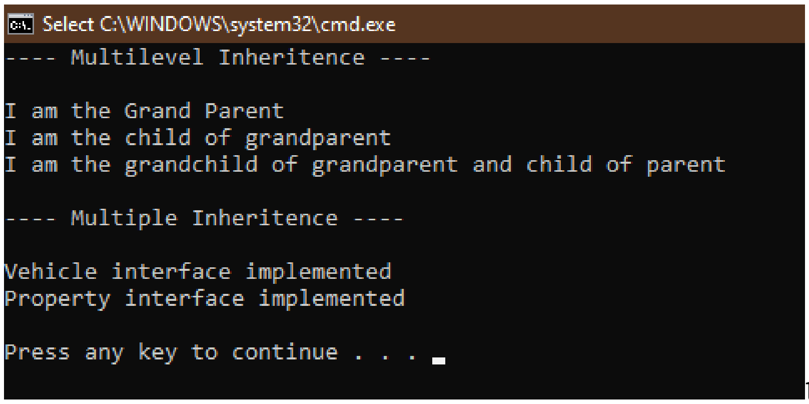


```
tNet                                                    DotNet.Parent
using System;

namespace DotNet
{
    1 reference
    interface Vehicle
    {
        2 references
        void car();
    }

    1 reference
    interface Property
    {
        2 references
        void house();
    }

    1 reference
    internal class GrandParent
    {
        1 reference
        public void GrandParentIdentity()
        {
            Console.WriteLine( "I am the Grand Parent" );
        }
    }

    1 reference
    internal class Parent : GrandParent
    {
        1 reference
        public void ParentIdentity()
        {
            Console.WriteLine("I am the child of grandparent");
        }
    }
```



```
2 references
internal class Child : Parent
{
    1 reference
    public void ChildIdentity()
    {
        Console.WriteLine("I am the grandchild of grandparent and child of parent");
    }

    0 references
    static void Main(string[] args)
    {
        Console.WriteLine( "---- Multilevel Inheritence ----" );
        Console.WriteLine();
        Child obj = new Child();
        obj.GrandParentIdentity();
        obj.ParentIdentity();
        obj.ChildIdentity();
        Console.WriteLine();

        Console.WriteLine("---- Multiple Inheritence ----");
        Console.WriteLine();
        new InterfaceImplementation();
        Console.WriteLine();
    }
}

2 references
internal class InterfaceImplementation : Vehicle, Property
{
    1 reference
    public InterfaceImplementation()
    {
        car();
        house();
    }
    2 references
    public void car()
    {
        Console.WriteLine( "Vehicle interface implemented" );
    }

    2 references
    public void house()
    {
        Console.WriteLine( "Property interface implemented" );
    }
}
}
```

Result



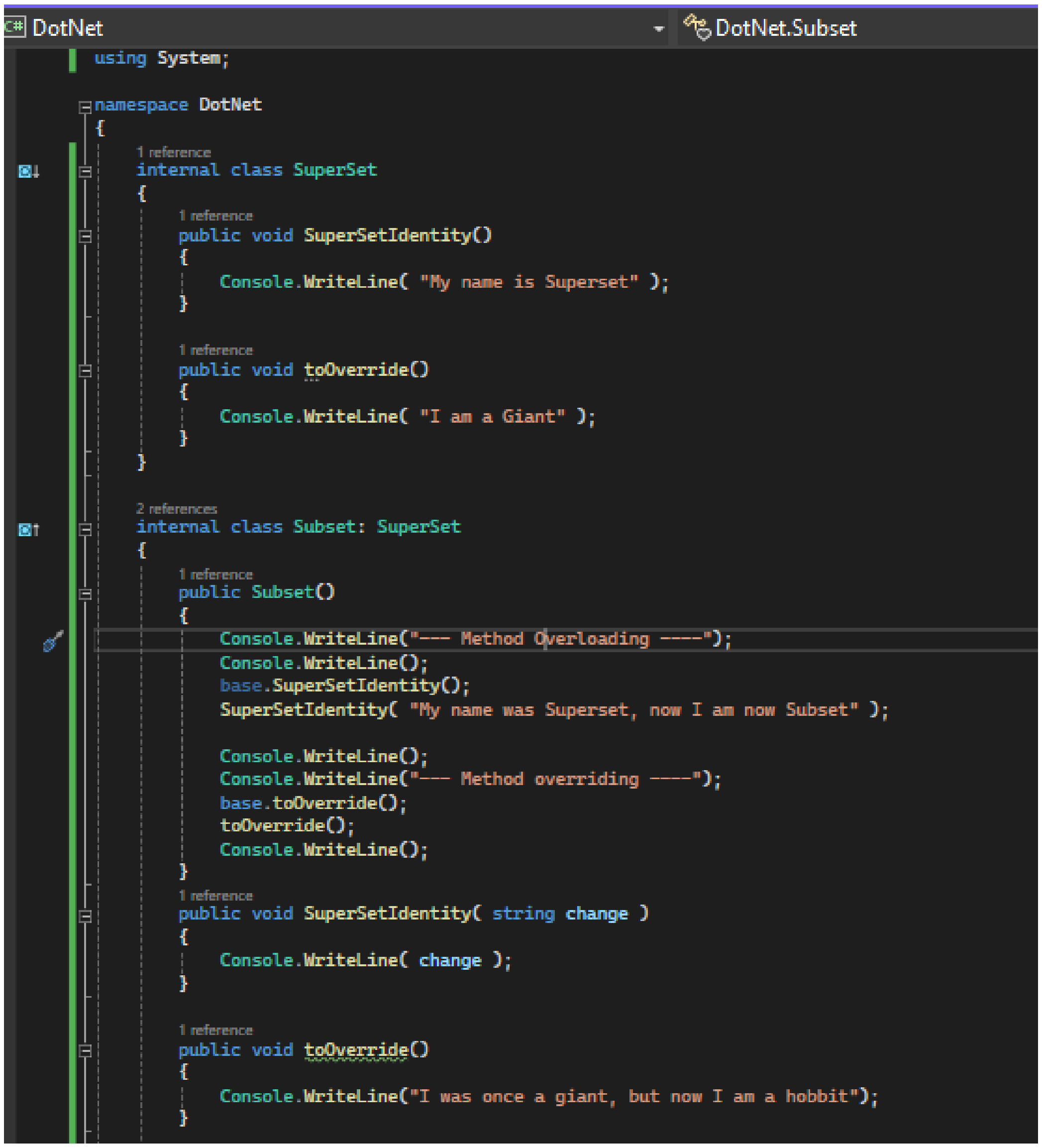
```
---- Multilevel Inheritence ----

I am the Grand Parent
I am the child of grandparent
I am the grandchild of grandparent and child of parent

---- Multiple Inheritence ----

Vehicle interface implemented
Property interface implemented

Press any key to continue . . .
```

t

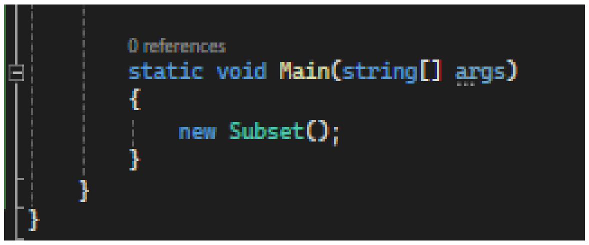
Problem – 4

Write a program to on method overloading and method overriding in c#

Code

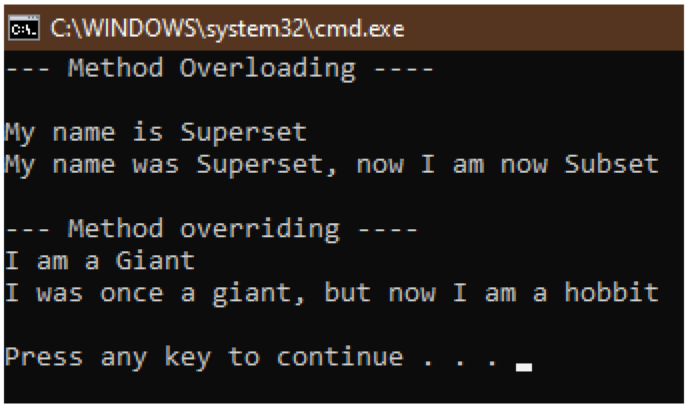


```
using System;

namespace DotNet
{
    1 reference
    internal class SuperSet
    {
        1 reference
        public void SuperSetIdentity()
        {
            Console.WriteLine( "My name is Superset" );
        }

        1 reference
        public void toOverride()
        {
            Console.WriteLine( "I am a Giant" );
        }
    }

    2 references
    internal class Subset: SuperSet
    {
        1 reference
        public Subset()
        {
            Console.WriteLine("--- Method Overloading ----");
            Console.WriteLine();
            base.SuperSetIdentity();
            SuperSetIdentity( "My name was Superset, now I am now Subset" );

            Console.WriteLine();
            Console.WriteLine("--- Method overriding ----");
            base.toOverride();
            toOverride();
            Console.WriteLine();
        }
        1 reference
        public void SuperSetIdentity( string change )
        {
            Console.WriteLine( change );
        }

        1 reference
        public void toOverride()
        {
            Console.WriteLine("I was once a giant, but now I am a hobbit");
        }
```



```
        0 references
        static void Main(string[] args)
        {
            new Subset();
        }
    }
}
```

Result

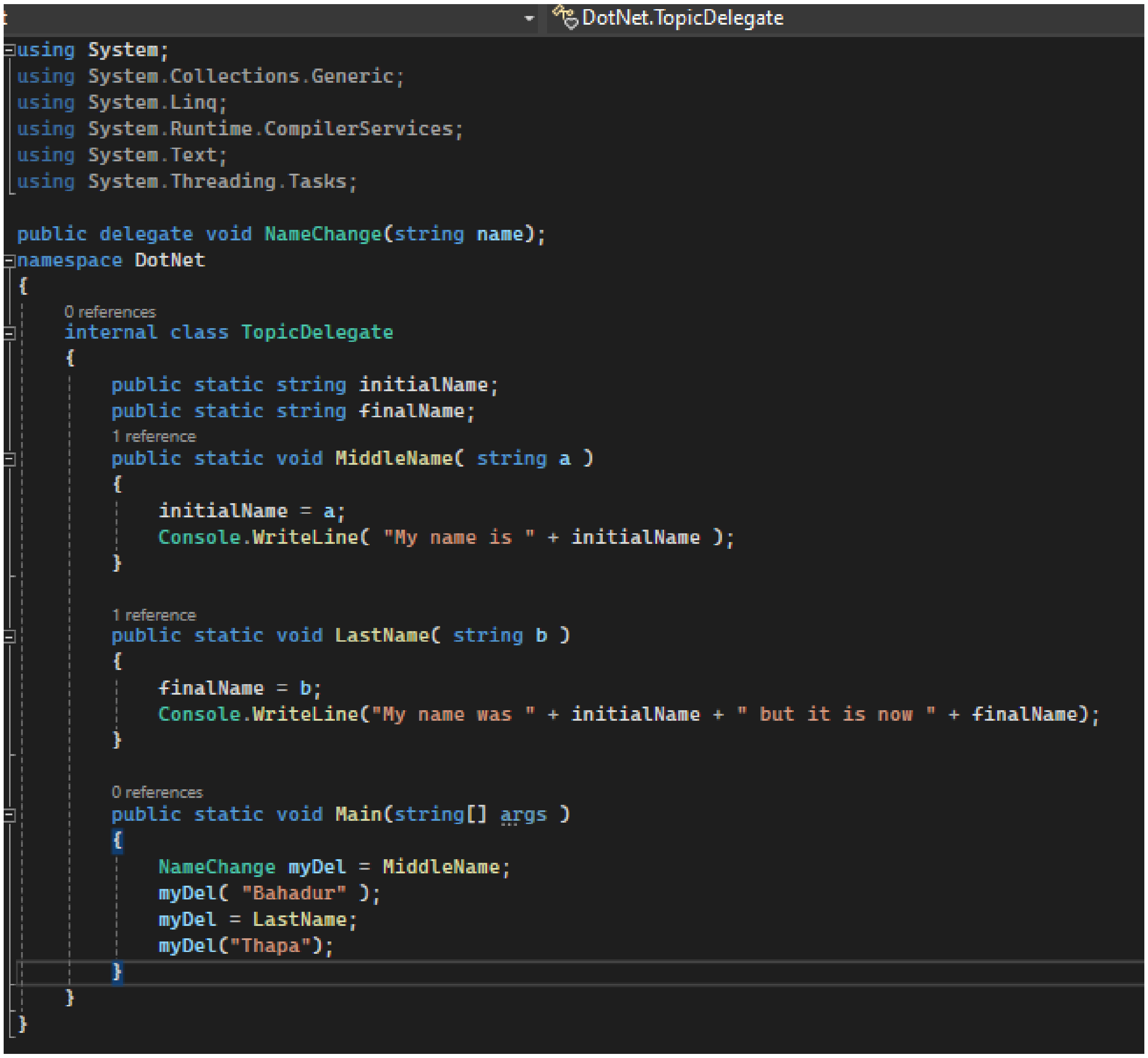


```
--- Method Overloading ----

My name is Superset
My name was Superset, now I am now Subset

--- Method overriding ----
I am a Giant
I was once a giant, but now I am a hobbit

Press any key to continue . . .
```

Lab – 5

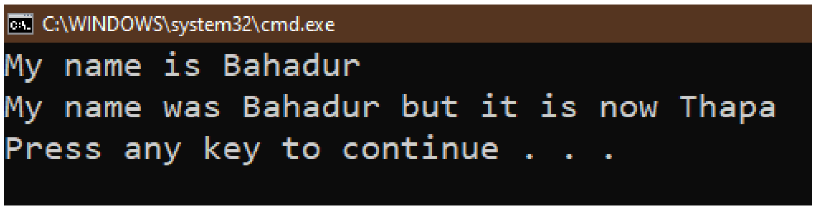
Write a program to demonstrate the concepts of Delegates

Code



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

public delegate void NameChange(string name);
namespace DotNet
{
    0 references
    internal class TopicDelegate
    {
        public static string initialName;
        public static string finalName;
        1 reference
        public static void MiddleName( string a )
        {
            initialName = a;
            Console.WriteLine( "My name is " + initialName );
        }

        1 reference
        public static void LastName( string b )
        {
            finalName = b;
            Console.WriteLine("My name was " + initialName + " but it is now " + finalName);
        }

        0 references
        public static void Main(string[] args )
        {
            NameChange myDel = MiddleName;
            myDel( "Bahadur" );
            myDel = LastName;
            myDel("Thapa");
        }
    }
}
```

Result

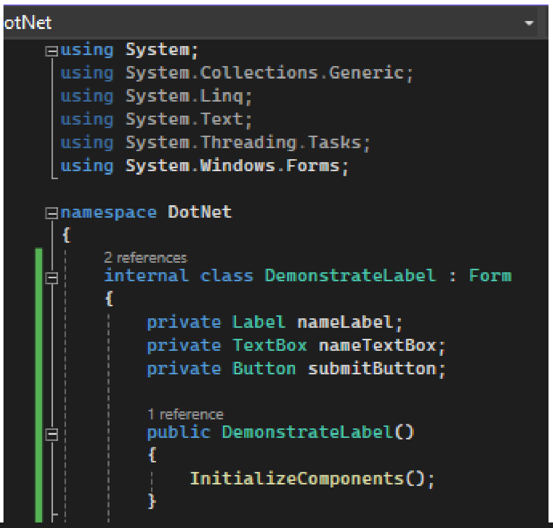


```
My name is Bahadur
My name was Bahadur but it is now Thapa
Press any key to continue . . .
```

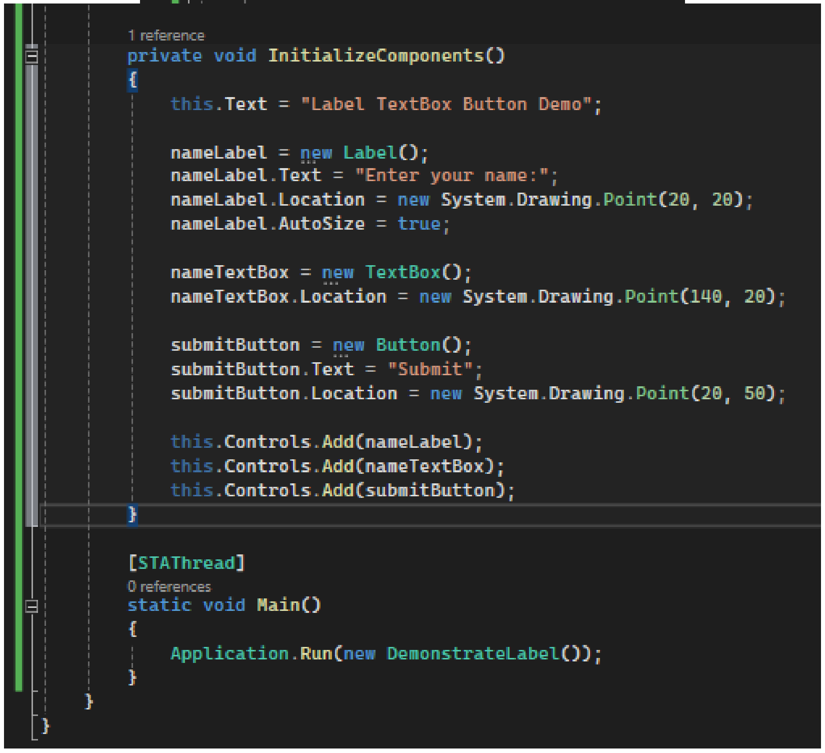
Lab – 6

Write a program to demonstrate the concepts of labels, text box and button controls.

Code

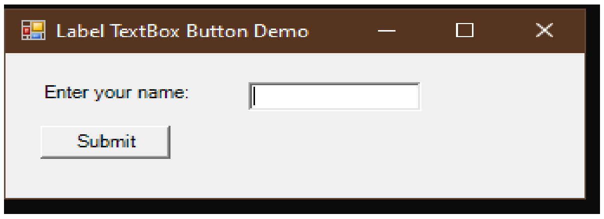


```
otNet
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace DotNet
{
    2 references
    internal class DemonstrateLabel : Form
    {
        private Label nameLabel;
        private TextBox nameTextBox;
        private Button submitButton;

        1 reference
        public DemonstrateLabel()
        {
            InitializeComponents();
        }
```



```
        1 reference
        private void InitializeComponents()
        {
            this.Text = "Label TextBox Button Demo";

            nameLabel = new Label();
            nameLabel.Text = "Enter your name:";
            nameLabel.Location = new System.Drawing.Point(20, 20);
            nameLabel.AutoSize = true;

            nameTextBox = new TextBox();
            nameTextBox.Location = new System.Drawing.Point(140, 20);

            submitButton = new Button();
            submitButton.Text = "Submit";
            submitButton.Location = new System.Drawing.Point(20, 50);

            this.Controls.Add(nameLabel);
            this.Controls.Add(nameTextBox);
            this.Controls.Add(submitButton);
        }

        [STAThread]
        0 references
        static void Main()
        {
            Application.Run(new DemonstrateLabel());
        }
    }
}
```

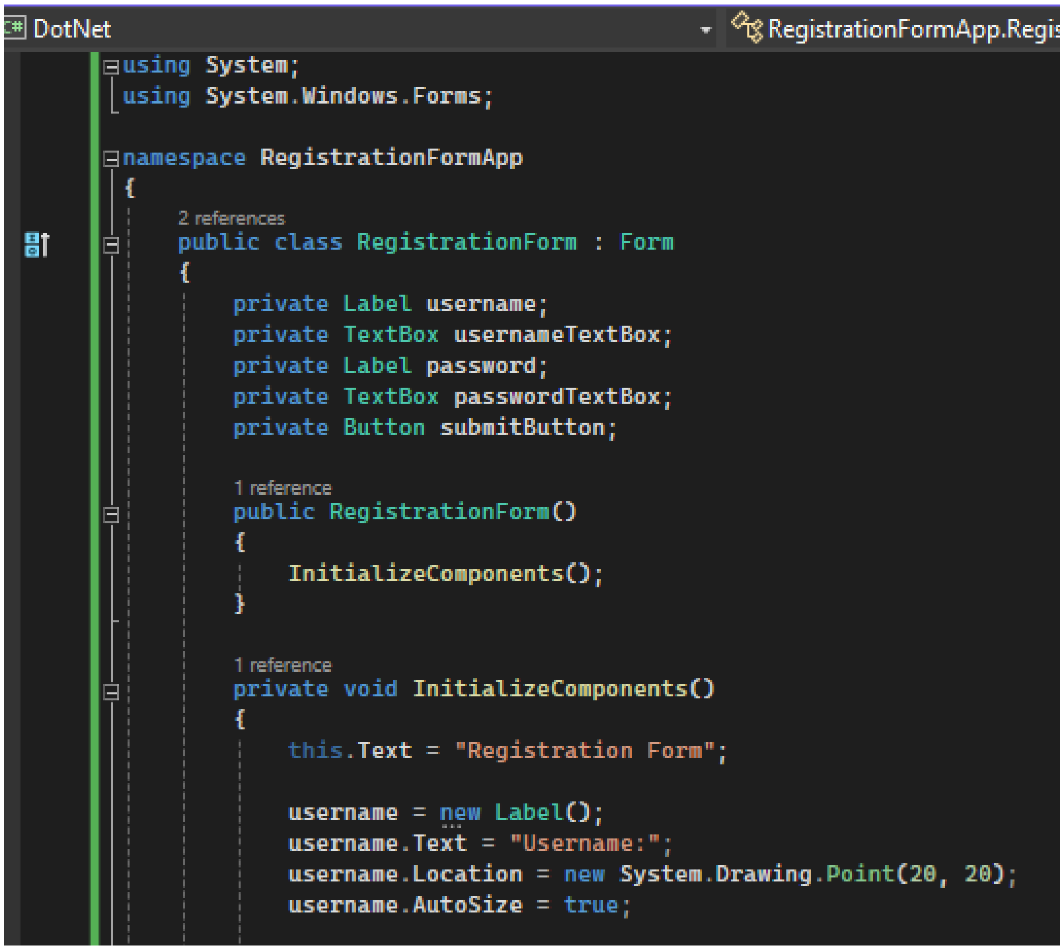
Result



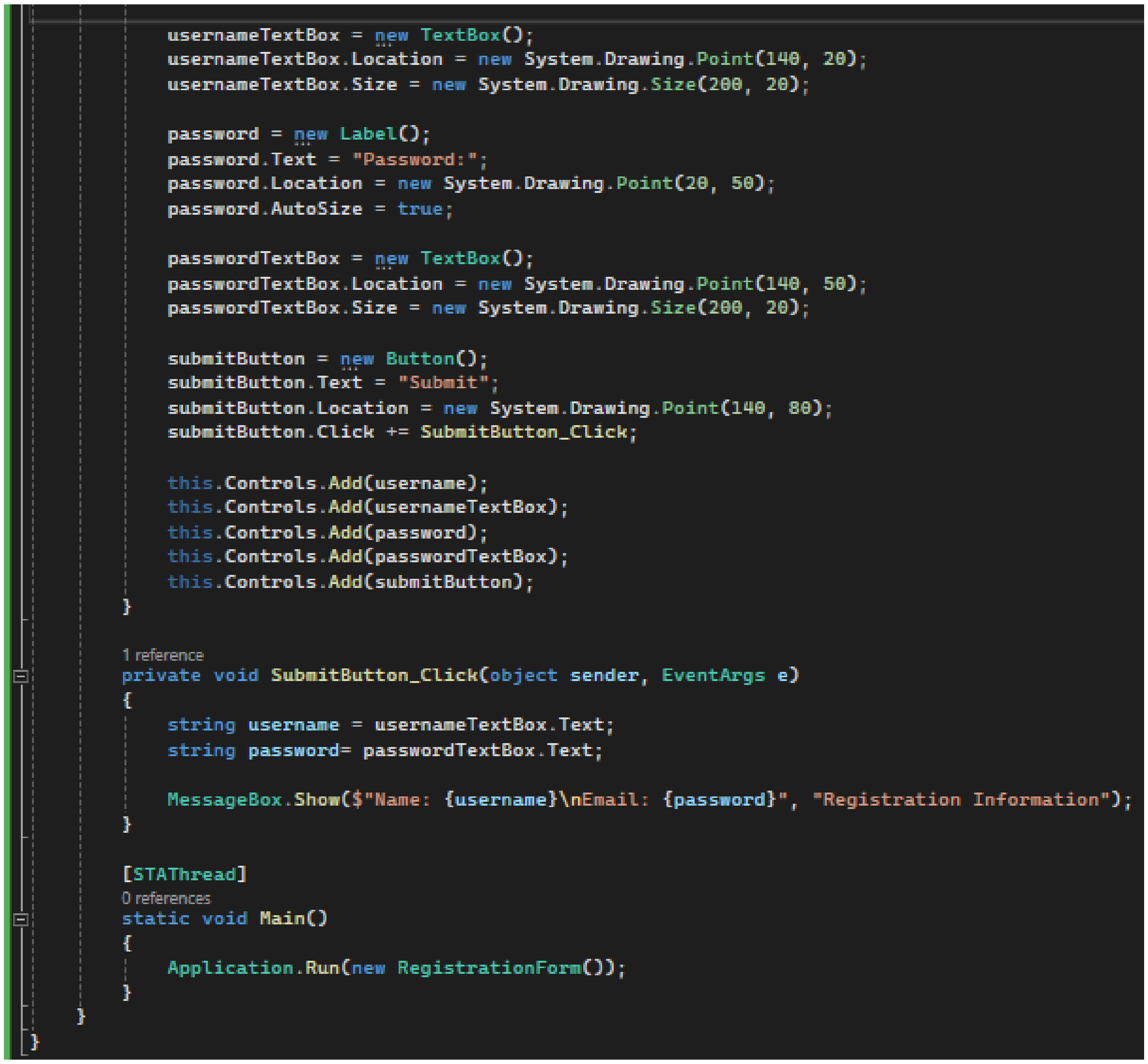
Lab – 7

Create a windows application in C# for registration form and fill the details and when you click the submit button it displays the details in the message box.

Code

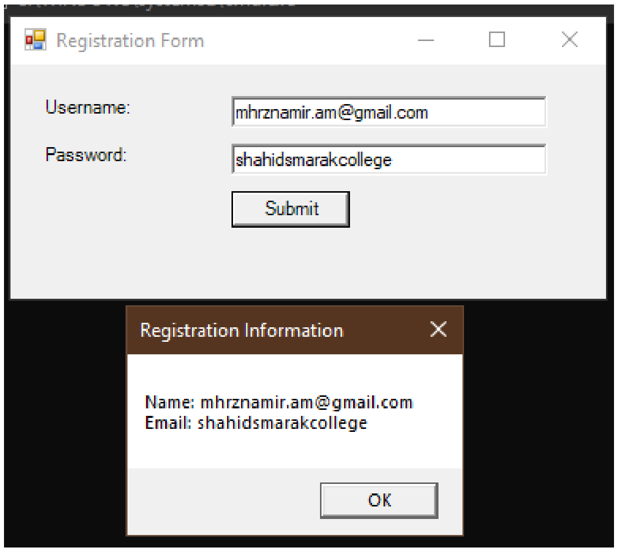


```
using System;
using System.Windows.Forms;

namespace RegistrationFormApp
{
    2 references
    public class RegistrationForm : Form
    {
        private Label username;
        private TextBox usernameTextBox;
        private Label password;
        private TextBox passwordTextBox;
        private Button submitButton;

        1 reference
        public RegistrationForm()
        {
            InitializeComponents();
        }

        1 reference
        private void InitializeComponents()
        {
            this.Text = "Registration Form";

            username = new Label();
            username.Text = "Username:";
            username.Location = new System.Drawing.Point(20, 20);
            username.AutoSize = true;
```



```
            usernameTextBox = new TextBox();
            usernameTextBox.Location = new System.Drawing.Point(140, 20);
            usernameTextBox.Size = new System.Drawing.Size(200, 20);

            password = new Label();
            password.Text = "Password:";
            password.Location = new System.Drawing.Point(20, 50);
            password.AutoSize = true;

            passwordTextBox = new TextBox();
            passwordTextBox.Location = new System.Drawing.Point(140, 50);
            passwordTextBox.Size = new System.Drawing.Size(200, 20);

            submitButton = new Button();
            submitButton.Text = "Submit";
            submitButton.Location = new System.Drawing.Point(140, 80);
            submitButton.Click += SubmitButton_Click;

            this.Controls.Add(username);
            this.Controls.Add(usernameTextBox);
            this.Controls.Add(password);
            this.Controls.Add(passwordTextBox);
            this.Controls.Add(submitButton);
        }

        1 reference
        private void SubmitButton_Click(object sender, EventArgs e)
        {
            string username = usernameTextBox.Text;
            string password= passwordTextBox.Text;

            MessageBox.Show($"Name: {username}\nEmail: {password}", "Registration Information");
        }

        [STAThread]
        0 references
        static void Main()
        {
            Application.Run(new RegistrationForm());
        }
    }
}
```

Result



Registration Form

Username: mhrznamir.am@gmail.com

Password: shahidsmarakcollege

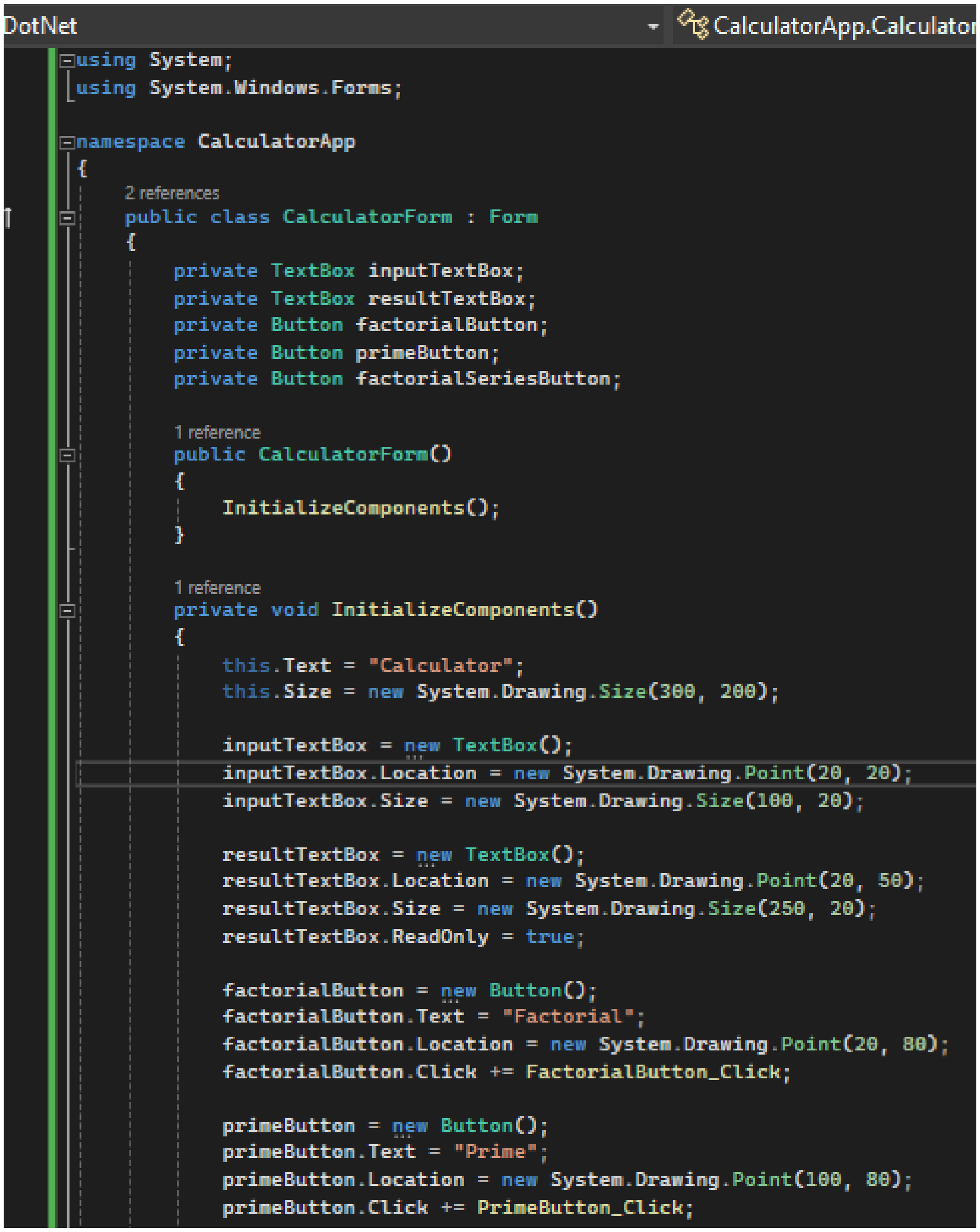
Registration Information

Name: mhrznamir.am@gmail.com
Email: shahidsmarakcollege

Lab – 8

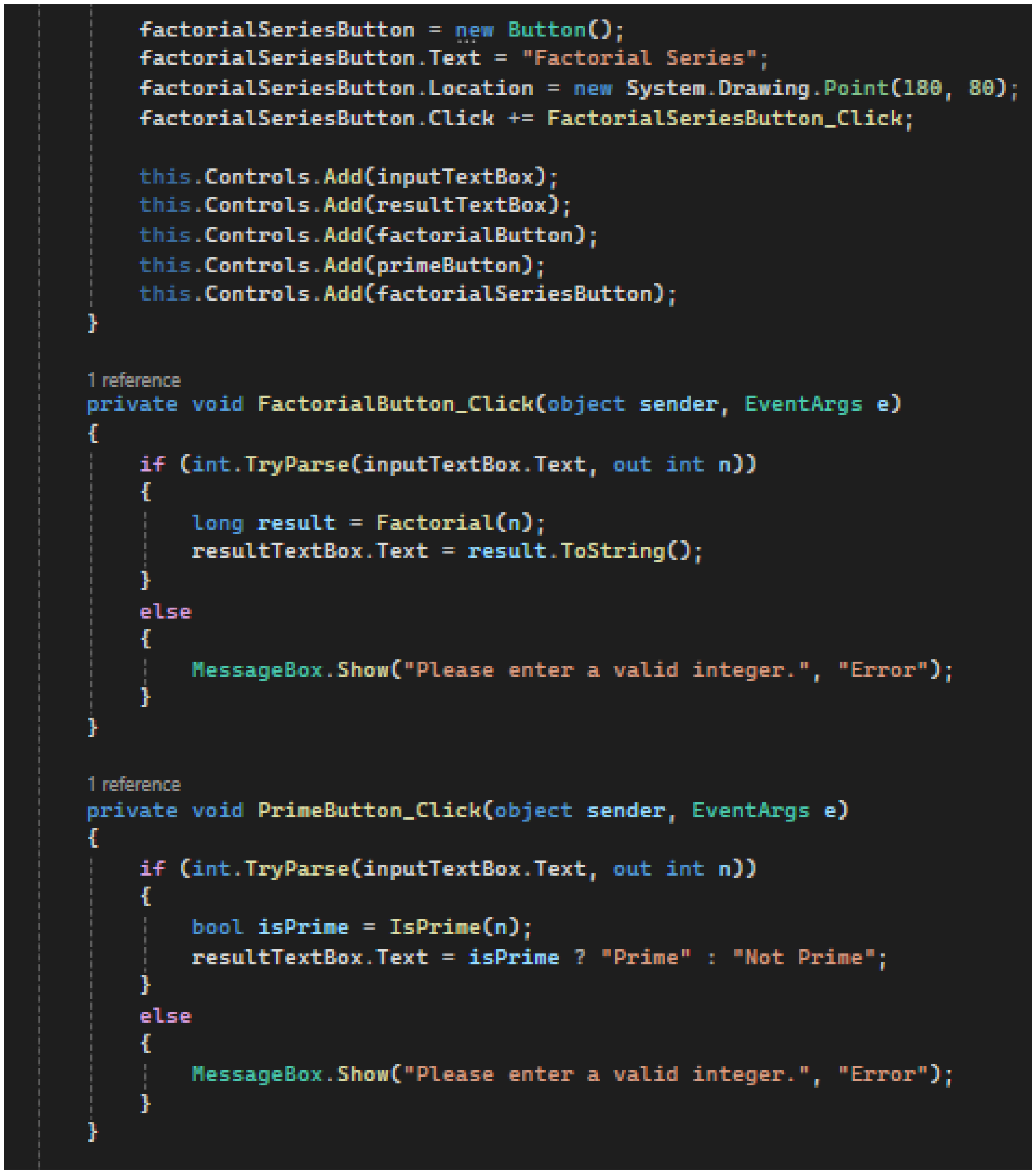
Create a Windows application in C# having two text boxes and three buttons named as factorial, prime, factorial series. When you click any button, the resultant value will be displayed on the second textbox.

Code

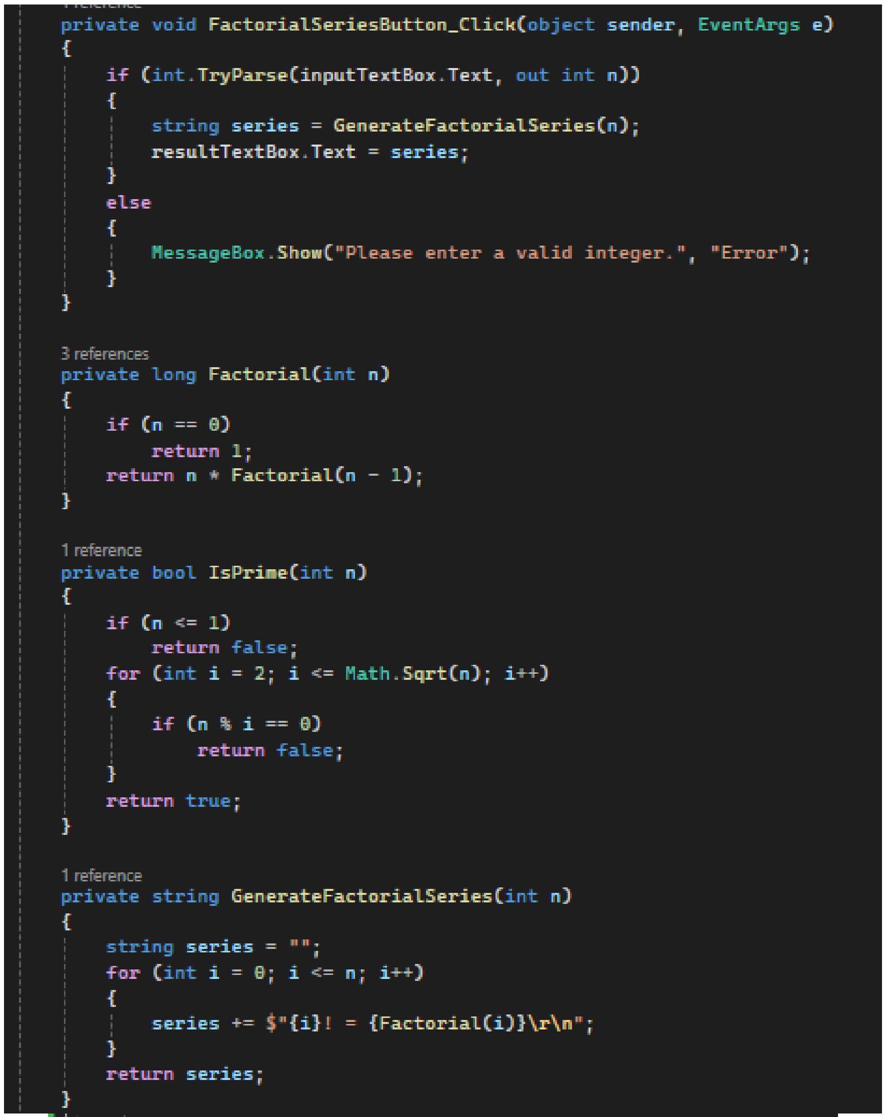


DotNet | CalculatorApp.Calculator

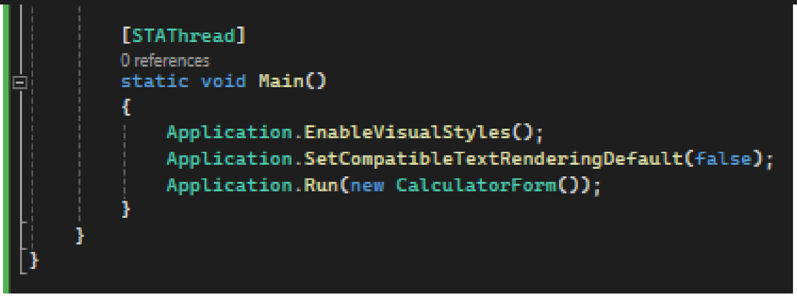
```
using System;
using System.Windows.Forms;

namespace CalculatorApp
{
    2 references
    public class CalculatorForm : Form
    {
        private TextBox inputTextBox;
        private TextBox resultTextBox;
        private Button factorialButton;
        private Button primeButton;
        private Button factorialSeriesButton;

        1 reference
        public CalculatorForm()
        {
            InitializeComponents();
        }

        1 reference
        private void InitializeComponents()
        {
            this.Text = "Calculator";
            this.Size = new System.Drawing.Size(300, 200);

            inputTextBox = new TextBox();
            inputTextBox.Location = new System.Drawing.Point(20, 20);
            inputTextBox.Size = new System.Drawing.Size(100, 20);

            resultTextBox = new TextBox();
            resultTextBox.Location = new System.Drawing.Point(20, 50);
            resultTextBox.Size = new System.Drawing.Size(250, 20);
            resultTextBox.ReadOnly = true;

            factorialButton = new Button();
            factorialButton.Text = "Factorial";
            factorialButton.Location = new System.Drawing.Point(20, 80);
            factorialButton.Click += FactorialButton_Click;

            primeButton = new Button();
            primeButton.Text = "Prime";
            primeButton.Location = new System.Drawing.Point(100, 80);
            primeButton.Click += PrimeButton_Click;
```



```
        factorialSeriesButton = new Button();
        factorialSeriesButton.Text = "Factorial Series";
        factorialSeriesButton.Location = new System.Drawing.Point(180, 80);
        factorialSeriesButton.Click += FactorialSeriesButton_Click;

        this.Controls.Add(inputTextBox);
        this.Controls.Add(resultTextBox);
        this.Controls.Add(factorialButton);
        this.Controls.Add(primeButton);
        this.Controls.Add(factorialSeriesButton);
    }

    1 reference
    private void FactorialButton_Click(object sender, EventArgs e)
    {
        if (int.TryParse(inputTextBox.Text, out int n))
        {
            long result = Factorial(n);
            resultTextBox.Text = result.ToString();
        }
        else
        {
            MessageBox.Show("Please enter a valid integer.", "Error");
        }
    }

    1 reference
    private void PrimeButton_Click(object sender, EventArgs e)
    {
        if (int.TryParse(inputTextBox.Text, out int n))
        {
            bool isPrime = IsPrime(n);
            resultTextBox.Text = isPrime ? "Prime" : "Not Prime";
        }
        else
        {
            MessageBox.Show("Please enter a valid integer.", "Error");
        }
    }
```

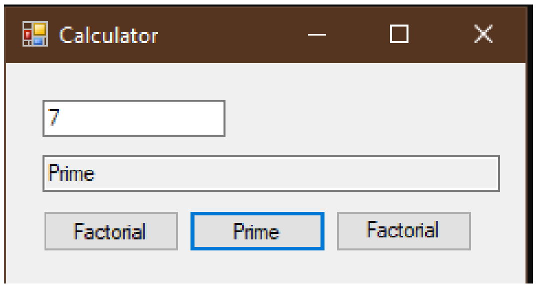


```
private void FactorialSeriesButton_Click(object sender, EventArgs e)
{
    if (int.TryParse(inputTextBox.Text, out int n))
    {
        string series = GenerateFactorialSeries(n);
        resultTextBox.Text = series;
    }
    else
    {
        MessageBox.Show("Please enter a valid integer.", "Error");
    }
}

3 references
private long Factorial(int n)
{
    if (n == 0)
        return 1;
    return n * Factorial(n - 1);
}

1 reference
private bool IsPrime(int n)
{
    if (n <= 1)
        return false;
    for (int i = 2; i <= Math.Sqrt(n); i++)
    {
        if (n % i == 0)
            return false;
    }
    return true;
}

1 reference
private string GenerateFactorialSeries(int n)
{
    string series = "";
    for (int i = 0; i <= n; i++)
    {
        series += $"{i}! = {Factorial(i)}\r\n";
    }
    return series;
}
```



```
        [STAThread]
        0 references
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new CalculatorForm());
        }
    }
}
```

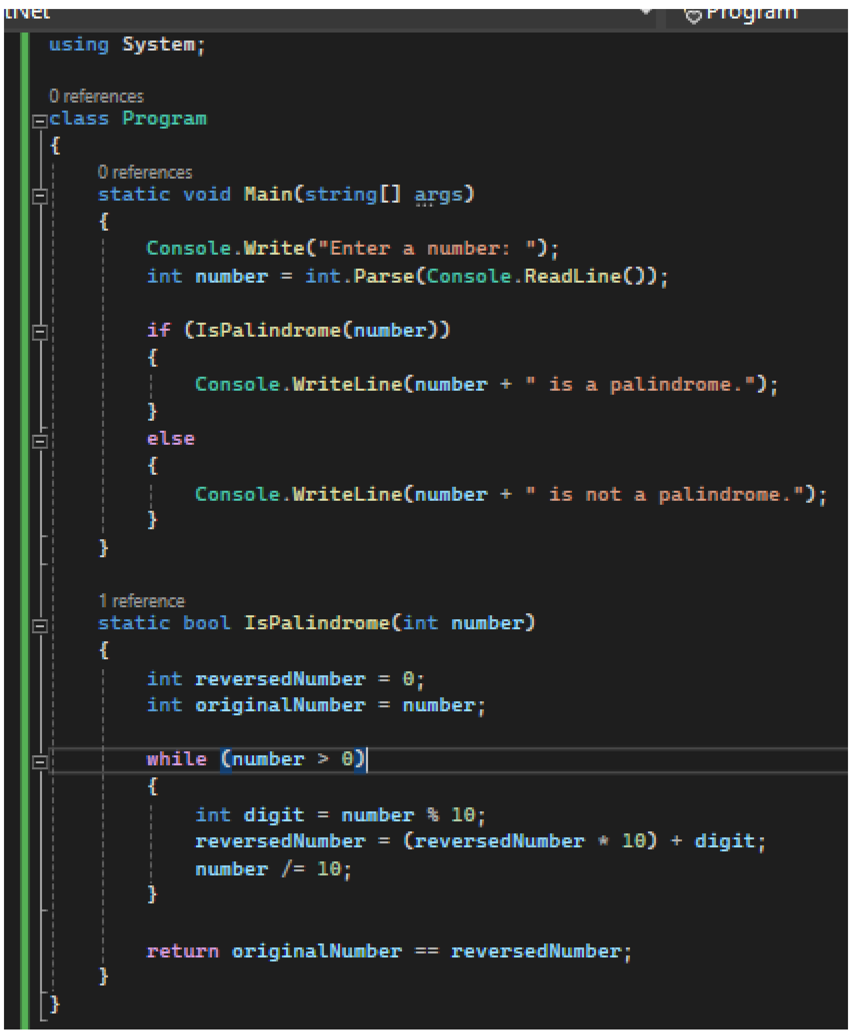
Result



Lab – 9

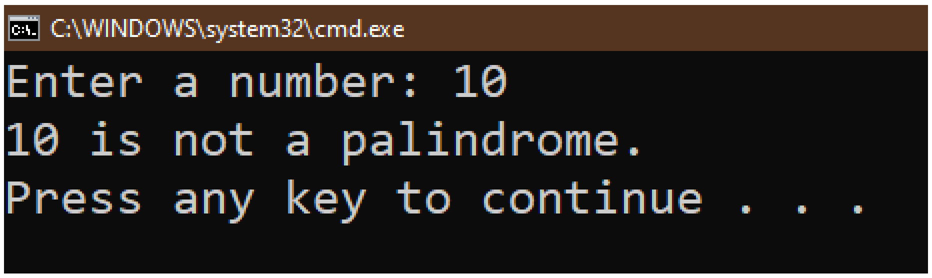
Write a program to check whether the number is palindrome or not.

Code



```
using System;

0 references
class Program
{
    0 references
    static void Main(string[] args)
    {
        Console.Write("Enter a number: ");
        int number = int.Parse(Console.ReadLine());

        if (IsPalindrome(number))
        {
            Console.WriteLine(number + " is a palindrome.");
        }
        else
        {
            Console.WriteLine(number + " is not a palindrome.");
        }
    }

    1 reference
    static bool IsPalindrome(int number)
    {
        int reversedNumber = 0;
        int originalNumber = number;

        while (number > 0)
        {
            int digit = number % 10;
            reversedNumber = (reversedNumber * 10) + digit;
            number /= 10;
        }

        return originalNumber == reversedNumber;
    }
}
```

Result

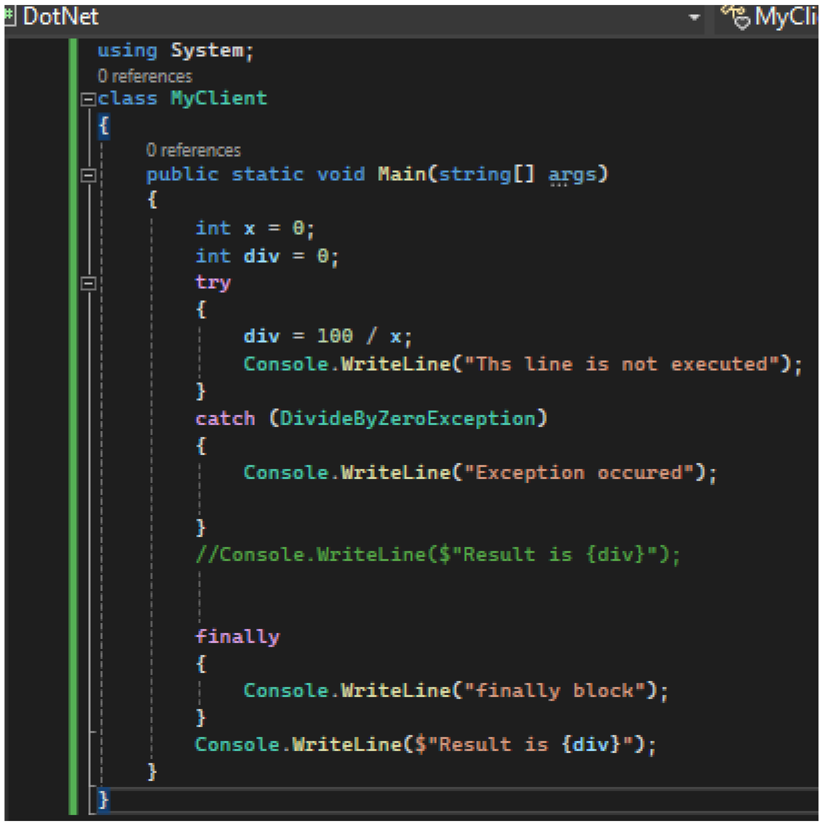


```
Enter a number: 10
10 is not a palindrome.
Press any key to continue . . .
```

Lab – 10

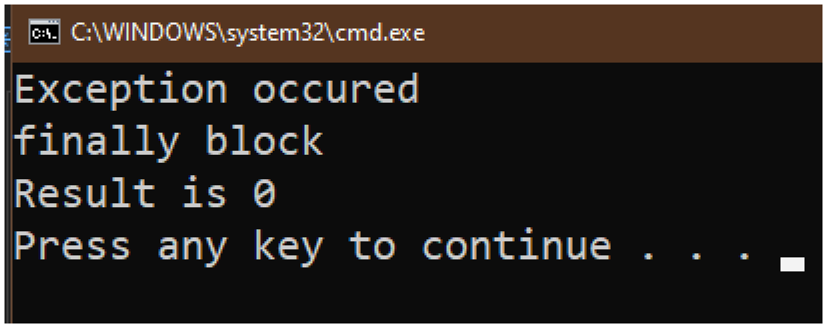
Demonstrate exception handling.

Code



```
using System;
0 references
class MyClient
{
    0 references
    public static void Main(string[] args)
    {
        int x = 0;
        int div = 0;
        try
        {
            div = 100 / x;
            Console.WriteLine("Ths line is not executed");
        }
        catch (DivideByZeroException)
        {
            Console.WriteLine("Exception occured");

        }
        //Console.WriteLine($"Result is {div}");


        finally
        {
            Console.WriteLine("finally block");
        }
        Console.WriteLine($"Result is {div}");
    }
}
```

Result

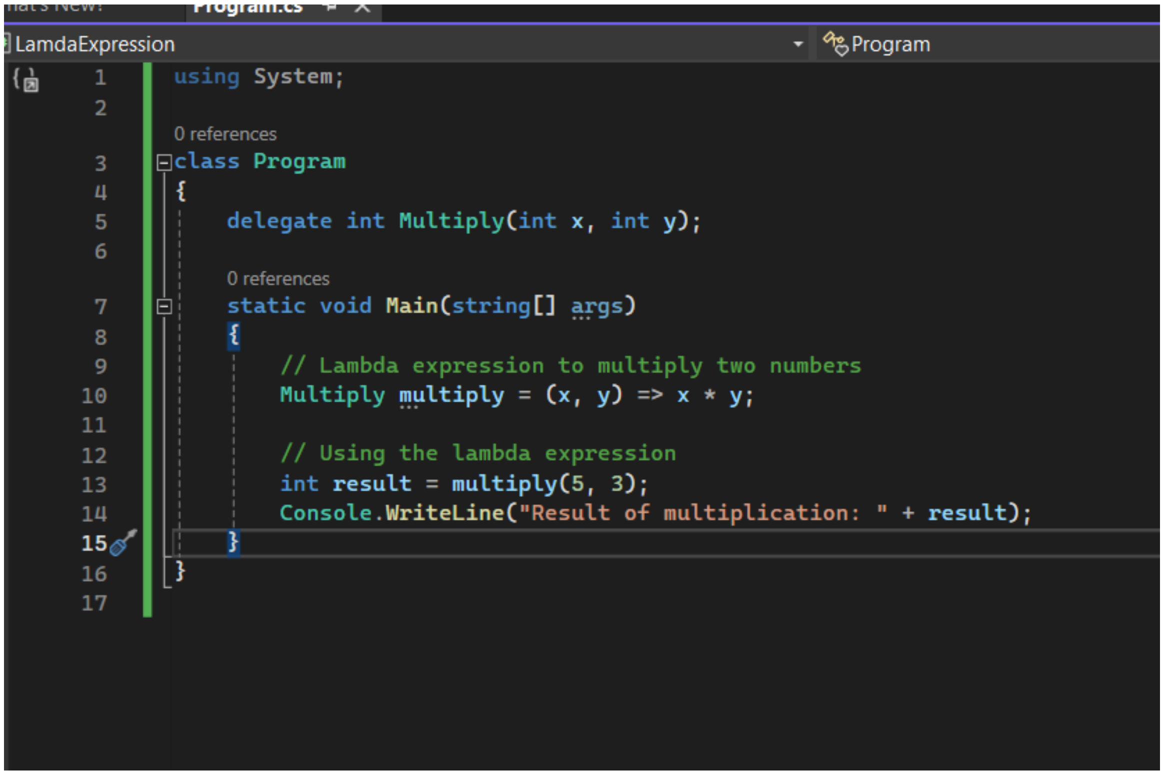


```
Exception occured
finally block
Result is 0
Press any key to continue . . .
```

Lab – 11

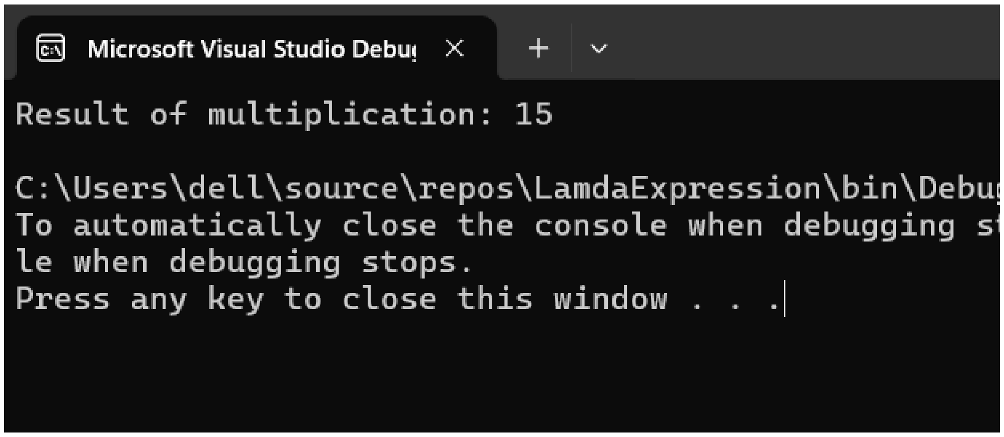
Write a program to implement lambda Expression

Code



```csharp
using System;

class Program
{
    delegate int Multiply(int x, int y);

    static void Main(string[] args)
    {
        // Lambda expression to multiply two numbers
        Multiply multiply = (x, y) => x * y;

        // Using the lambda expression
        int result = multiply(5, 3);
        Console.WriteLine("Result of multiplication: " + result);
    }
}
```

Result

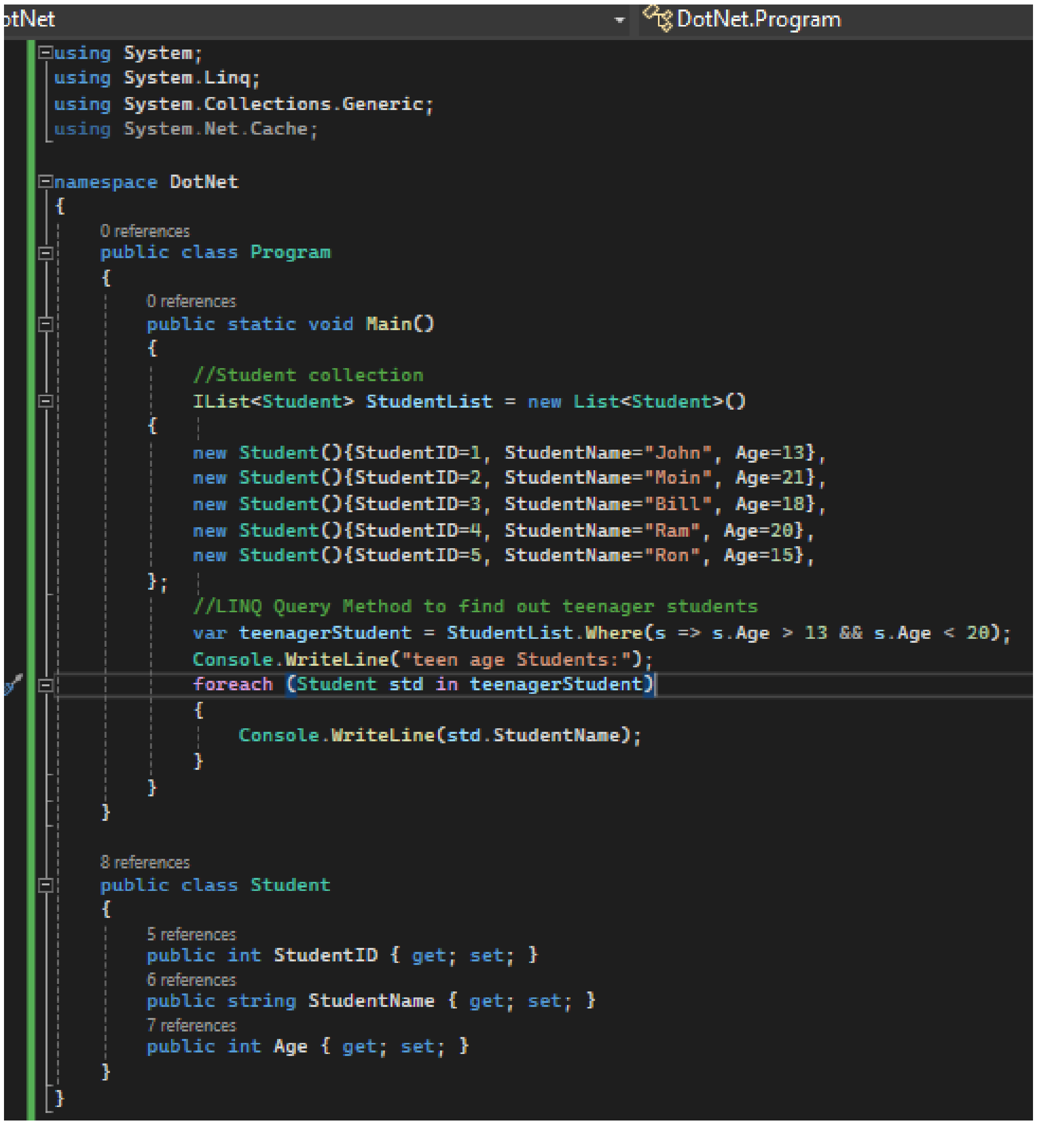


```
Result of multiplication: 15

C:\Users\dell\source\repos\LamdaExpression\bin\Debu
To automatically close the console when debugging st
le when debugging stops.
Press any key to close this window . . .
```

Lab – 12

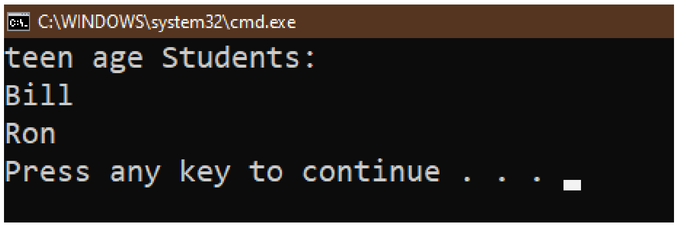
Write a program to check whether the number is palindrome or not.

Code



```
using System;
using System.Linq;
using System.Collections.Generic;
using System.Net.Cache;

namespace DotNet
{
    0 references
    public class Program
    {
        0 references
        public static void Main()
        {
            //Student collection
            IList<Student> StudentList = new List<Student>()
        {
            new Student(){StudentID=1, StudentName="John", Age=13},
            new Student(){StudentID=2, StudentName="Moin", Age=21},
            new Student(){StudentID=3, StudentName="Bill", Age=18},
            new Student(){StudentID=4, StudentName="Ram", Age=20},
            new Student(){StudentID=5, StudentName="Ron", Age=15},
        };
            //LINQ Query Method to find out teenager students
            var teenagerStudent = StudentList.Where(s => s.Age > 13 && s.Age < 20);
            Console.WriteLine("teen age Students:");
            foreach (Student std in teenagerStudent)
            {
                Console.WriteLine(std.StudentName);
            }
        }
    }

    8 references
    public class Student
    {
        5 references
        public int StudentID { get; set; }
        6 references
        public string StudentName { get; set; }
        7 references
        public int Age { get; set; }
    }
}
```

Result

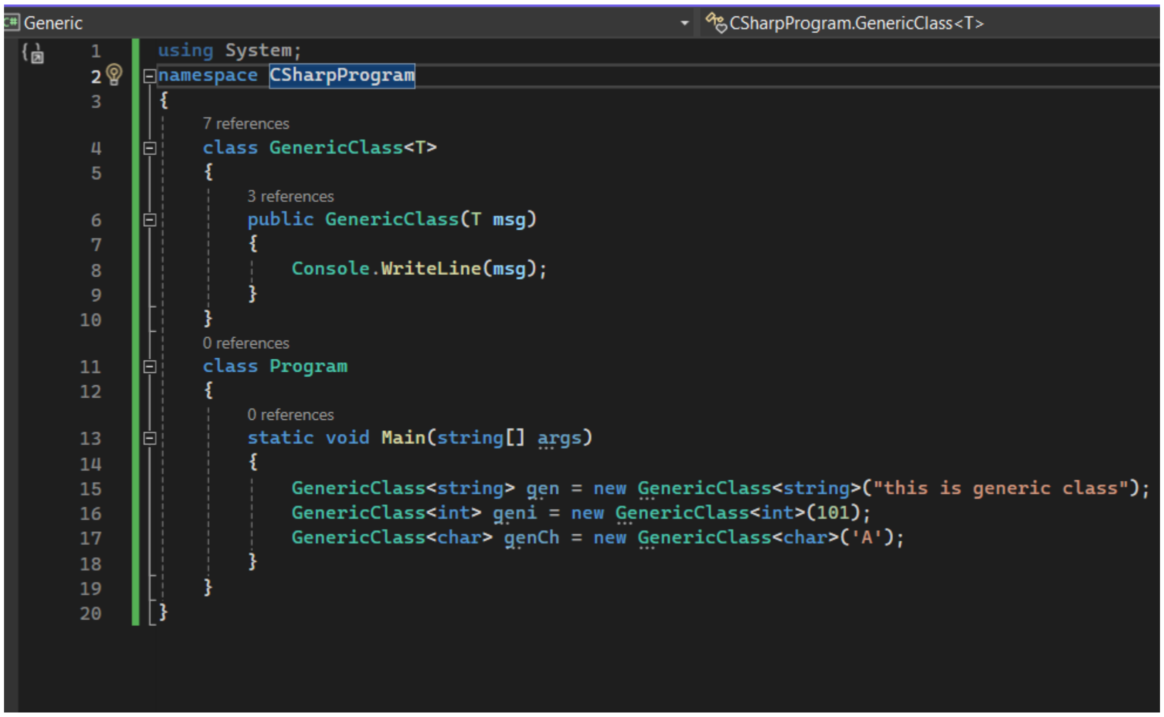


```
teen age Students:
Bill
Ron
Press any key to continue . . .
```

Lab – 13

Write a program to implement Generic class.

Code

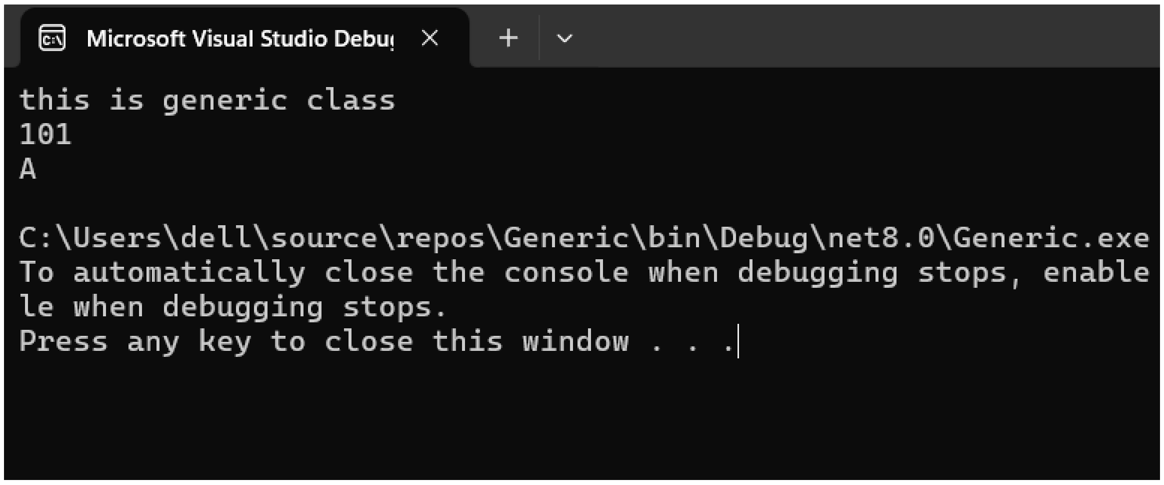


Generic

CSharpProgram.GenericClass<T>

```
using System;
namespace CSharpProgram
{
    7 references
    class GenericClass<T>
    {
        3 references
        public GenericClass(T msg)
        {
            Console.WriteLine(msg);
        }
    }
    0 references
    class Program
    {
        0 references
        static void Main(string[] args)
        {
            GenericClass<string> gen = new GenericClass<string>("this is generic class");
            GenericClass<int> geni = new GenericClass<int>(101);
            GenericClass<char> genCh = new GenericClass<char>('A');
        }
    }
}
```

Result



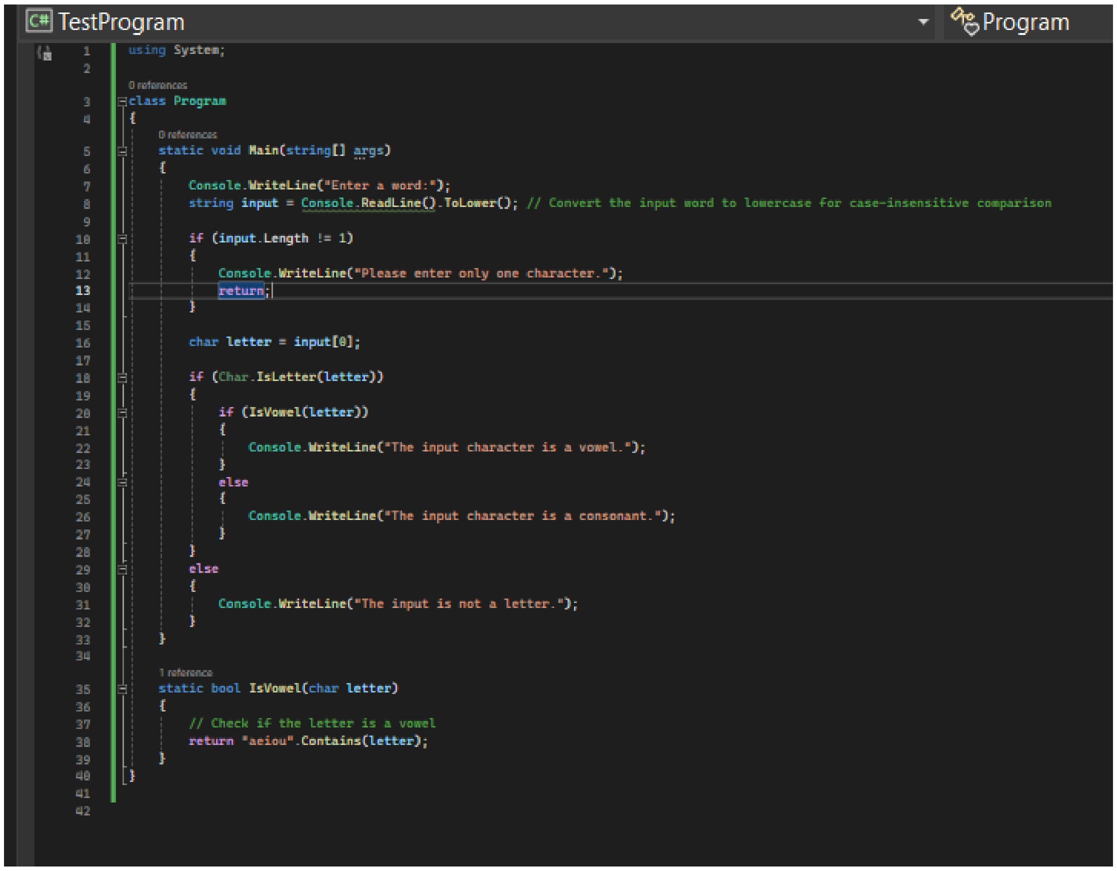
Microsoft Visual Studio Debu

```
this is generic class
101
A

C:\Users\dell\source\repos\Generic\bin\Debug\net8.0\Generic.exe
To automatically close the console when debugging stops, enable
le when debugging stops.
Press any key to close this window . . .
```

Lab – 14

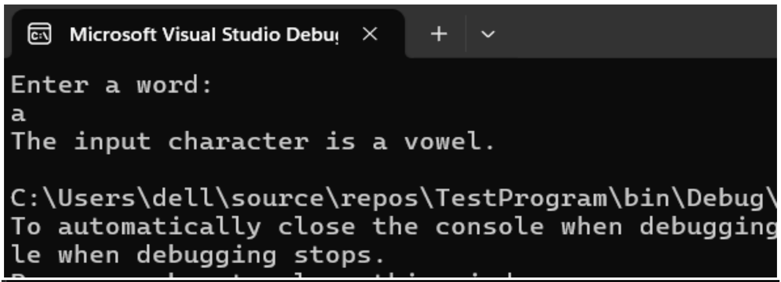
WAP to find whether the input word is vowel or consonant.

Code

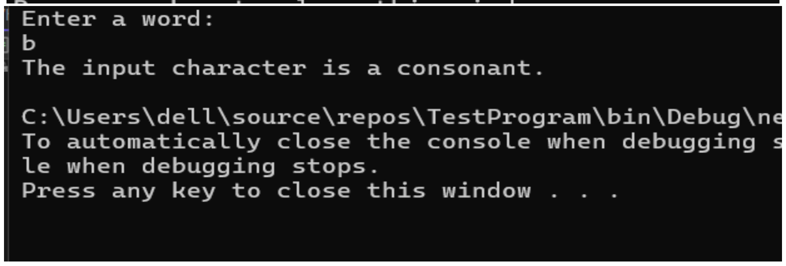


```csharp
using System;

class Program
{
    static void Main(string[] args)
    {
        Console.WriteLine("Enter a word:");
        string input = Console.ReadLine().ToLower(); // Convert the input word to lowercase for case-insensitive comparison

        if (input.Length != 1)
        {
            Console.WriteLine("Please enter only one character.");
            return;
        }

        char letter = input[0];

        if (Char.IsLetter(letter))
        {
            if (IsVowel(letter))
            {
                Console.WriteLine("The input character is a vowel.");
            }
            else
            {
                Console.WriteLine("The input character is a consonant.");
            }
        }
        else
        {
            Console.WriteLine("The input is not a letter.");
        }
    }

    static bool IsVowel(char letter)
    {
        // Check if the letter is a vowel
        return "aeiou".Contains(letter);
    }
}
```

Result



```
Enter a word:
a
The input character is a vowel.

C:\Users\dell\source\repos\TestProgram\bin\Debug\
To automatically close the console when debugging
le when debugging stops.
```

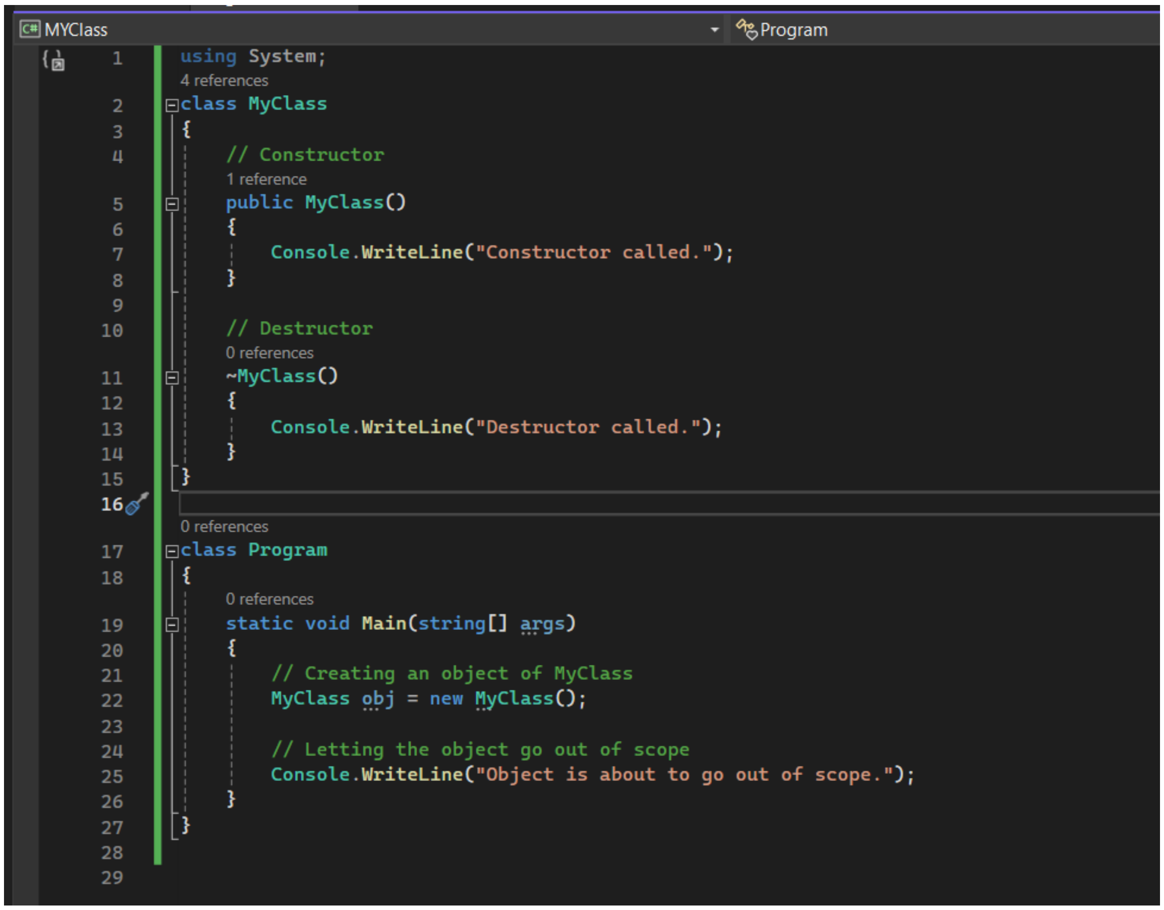


```
Enter a word:
b
The input character is a consonant.

C:\Users\dell\source\repos\TestProgram\bin\Debug\ne
To automatically close the console when debugging s
le when debugging stops.
Press any key to close this window . . .
```

Lab – 15

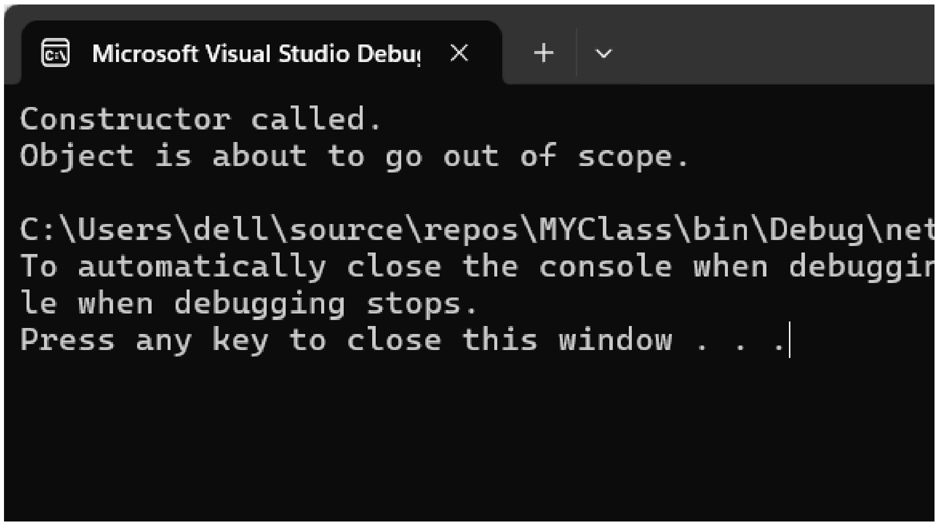
WAP to implement the concept of destructors.

Code



```
using System;
class MyClass
{
    // Constructor
    public MyClass()
    {
        Console.WriteLine("Constructor called.");
    }

    // Destructor
    ~MyClass()
    {
        Console.WriteLine("Destructor called.");
    }
}

class Program
{
    static void Main(string[] args)
    {
        // Creating an object of MyClass
        MyClass obj = new MyClass();

        // Letting the object go out of scope
        Console.WriteLine("Object is about to go out of scope.");
    }
}
```

Result

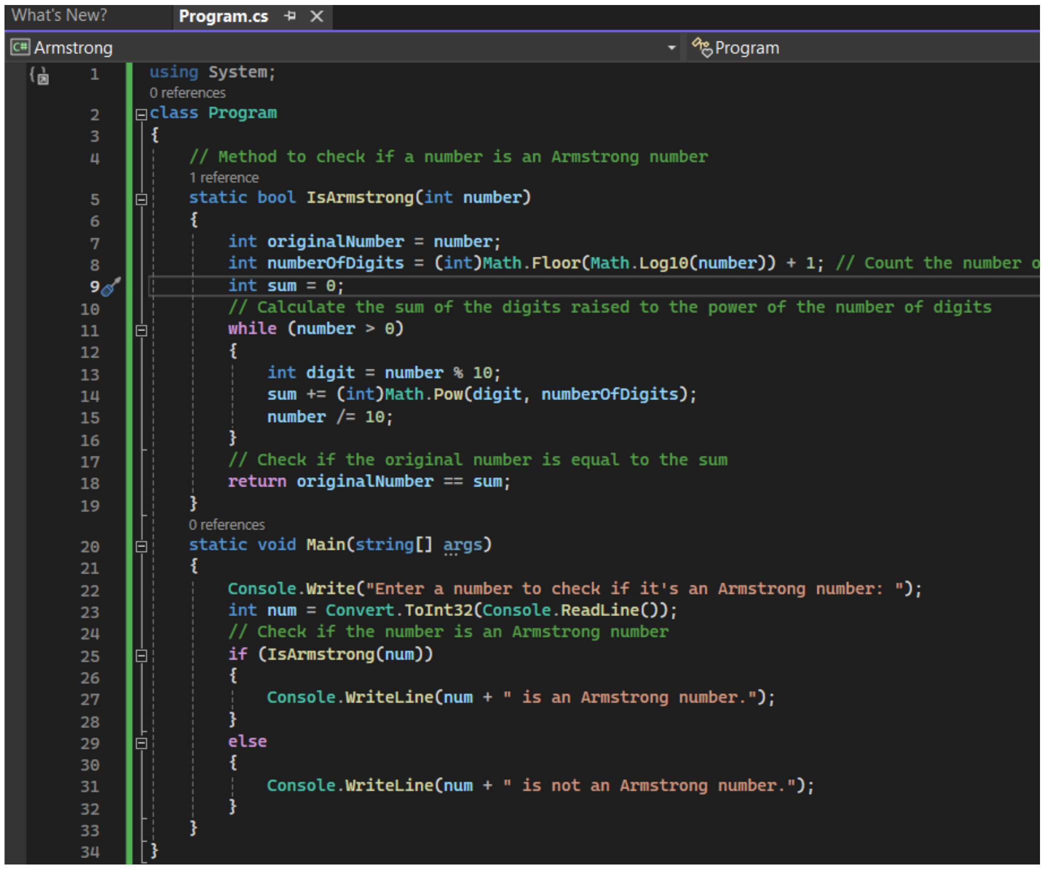


```
Constructor called.
Object is about to go out of scope.

C:\Users\dell\source\repos\MYClass\bin\Debug\net
To automatically close the console when debuggi
le when debugging stops.
Press any key to close this window . . .
```

Lab – 16

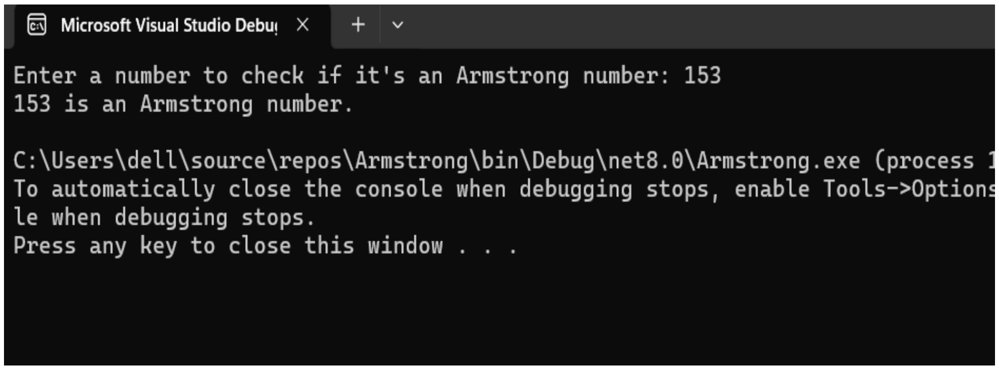
WAP to check a number whether it is Armstrong or not.

Code



```csharp
using System;
0 references
class Program
{
    // Method to check if a number is an Armstrong number
    1 reference
    static bool IsArmstrong(int number)
    {
        int originalNumber = number;
        int numberOfDigits = (int)Math.Floor(Math.Log10(number)) + 1; // Count the number o
        int sum = 0;
        // Calculate the sum of the digits raised to the power of the number of digits
        while (number > 0)
        {
            int digit = number % 10;
            sum += (int)Math.Pow(digit, numberOfDigits);
            number /= 10;
        }
        // Check if the original number is equal to the sum
        return originalNumber == sum;
    }
    0 references
    static void Main(string[] args)
    {
        Console.Write("Enter a number to check if it's an Armstrong number: ");
        int num = Convert.ToInt32(Console.ReadLine());
        // Check if the number is an Armstrong number
        if (IsArmstrong(num))
        {
            Console.WriteLine(num + " is an Armstrong number.");
        }
        else
        {
            Console.WriteLine(num + " is not an Armstrong number.");
        }
    }
}
```

Result



```
Enter a number to check if it's an Armstrong number: 153
153 is an Armstrong number.

C:\Users\dell\source\repos\Armstrong\bin\Debug\net8.0\Armstrong.exe (process 1
To automatically close the console when debugging stops, enable Tools->Options
le when debugging stops.
Press any key to close this window . . .
```