

Visual Programming Languages

Lab Manual

[Fall/Spring 2019]

|  |  |
| --- | --- |
| Student Name: Areesha Sohail |  |
| Student Id: 10601 |  |
|  |  |

|  |  |
| --- | --- |
| Prepared By:*Dr. Noman Islam* |  |
| Instructor: *Dr. Noman Islam* |  |

**LIST OF EXPERIMENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S. No** | **Date** | **Experiment** |  |
| **1** | \_\_/\_\_/\_\_ | To setup the environment and familiarize with C# |  |
| **2** | \_\_/\_\_/\_\_ | To study and understand how to write programs in C# using loops and arrays |  |
| **3** | \_\_/\_\_/\_\_ | To study and implement object oriented programming concepts in C# |  |
| **4** | \_\_/\_\_/\_\_ | To study and implement Windows Forms application in C# |  |
| **5** | \_\_/\_\_/\_\_ | To study and implement Collections in C# |  |
| **6** | \_\_/\_\_/\_\_ | To study and implement I/O in C# |  |
| **7** | \_\_/\_\_/\_\_ | To study and implement XML parsing in C# |  |
| **8** | \_\_/\_\_/\_\_ | To study and implement WPF and its layouts in C# |  |
| **9** | \_\_/\_\_/\_\_ | To study and implement LINQ in C# |  |
| **10** | \_\_/\_\_/\_\_ | To study and implement ADO .Net in C# |  |

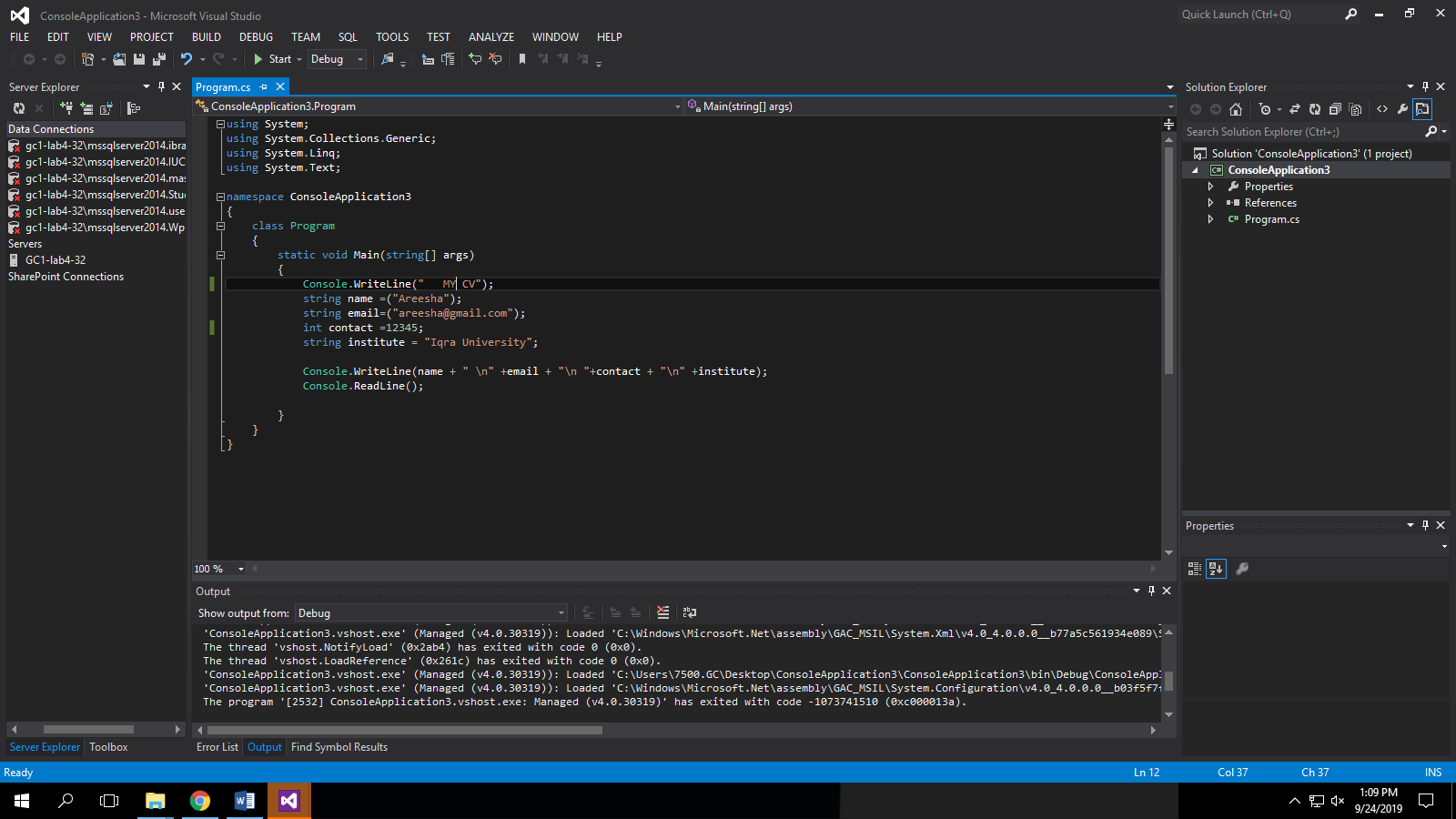
**Lab 1: To setup the environment and familiarize with C#**

The objective of this lab is to set up the Visual Studio environment and get some familiarity with the C# language.

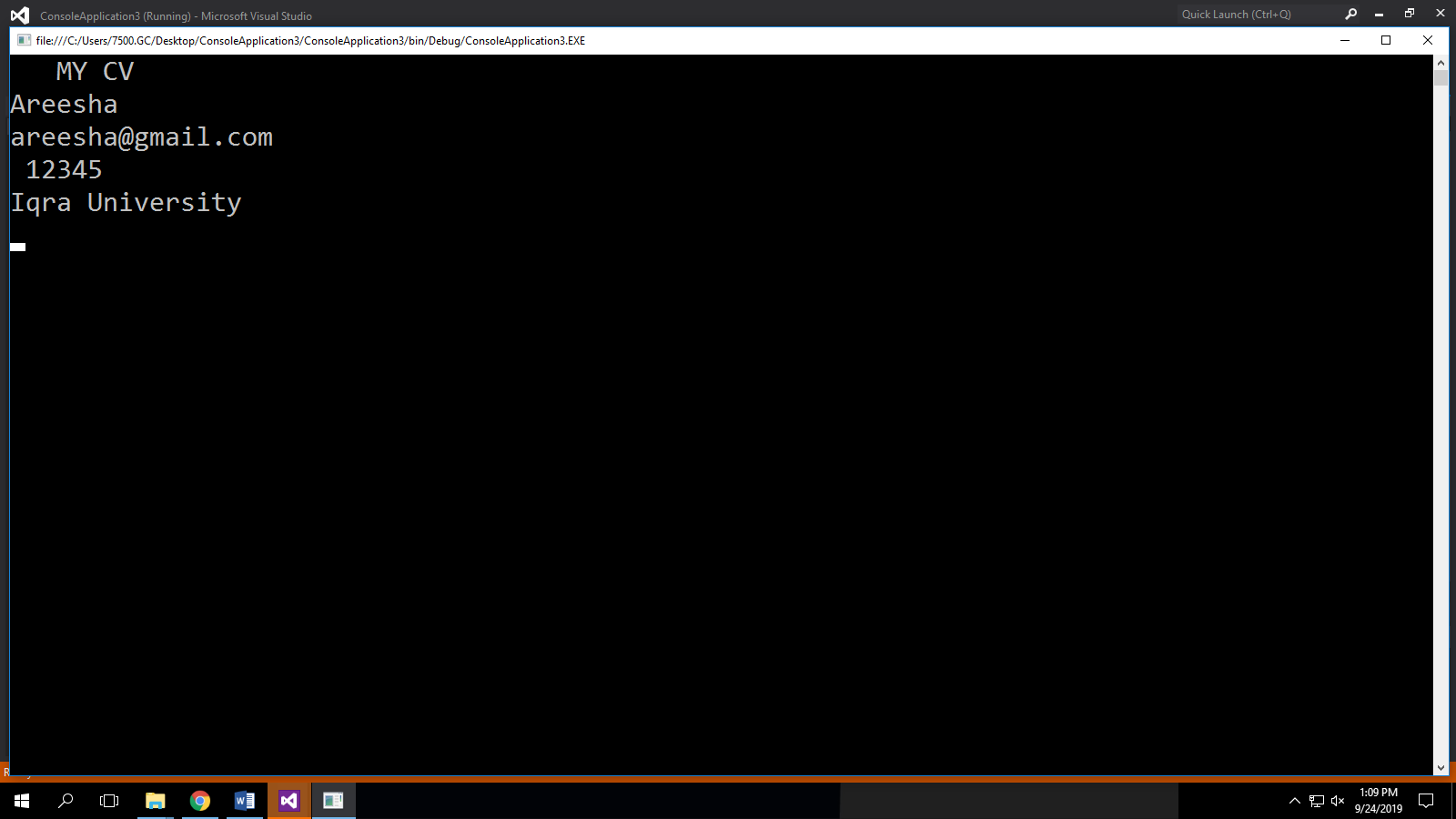
Download and install Visual Studio .Net. Visual Studio is the leading platform powered by Microsoft for development on .net framework

**Lab Tasks:**

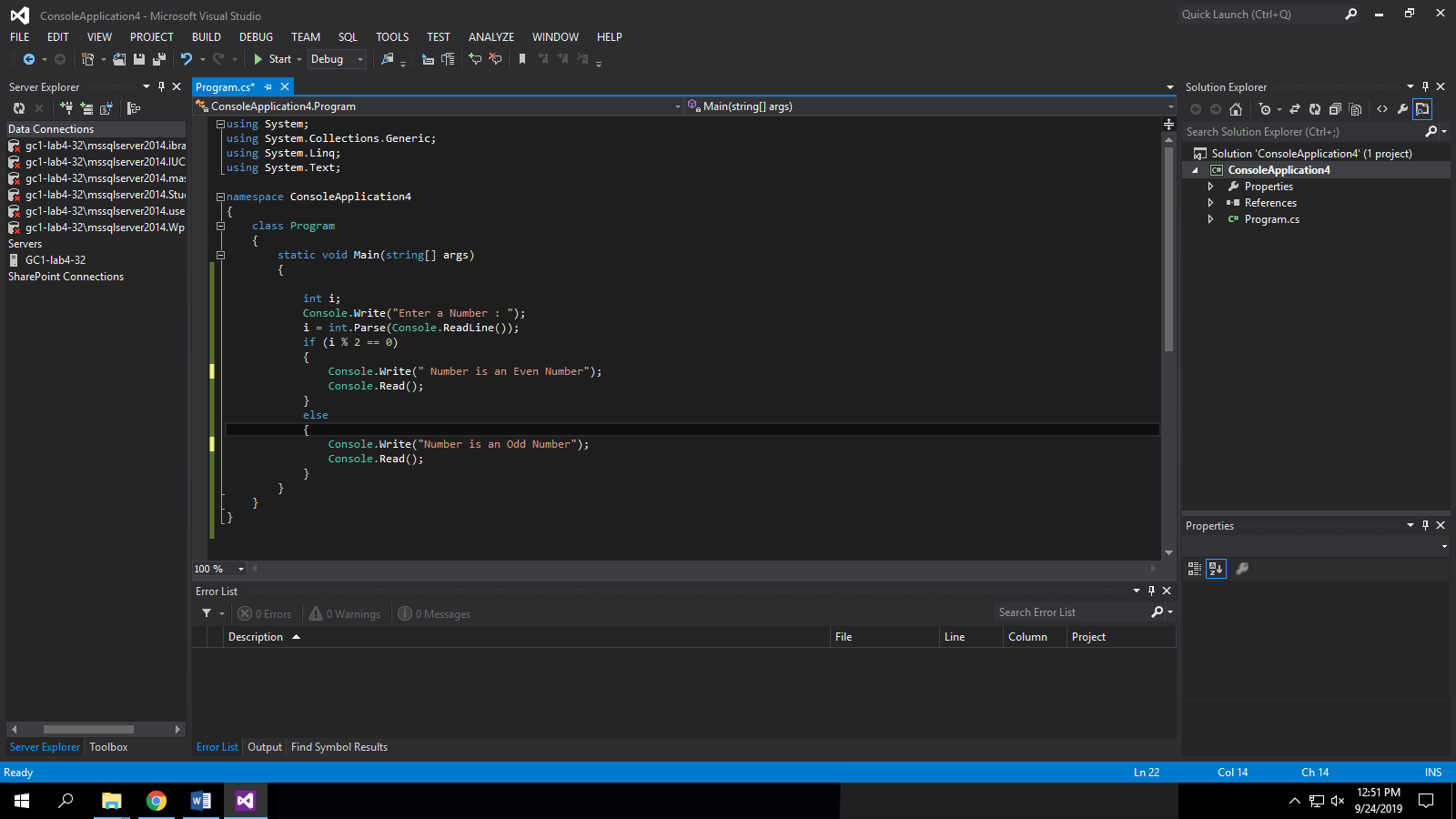
1. Write a small program in C# to print your CV.



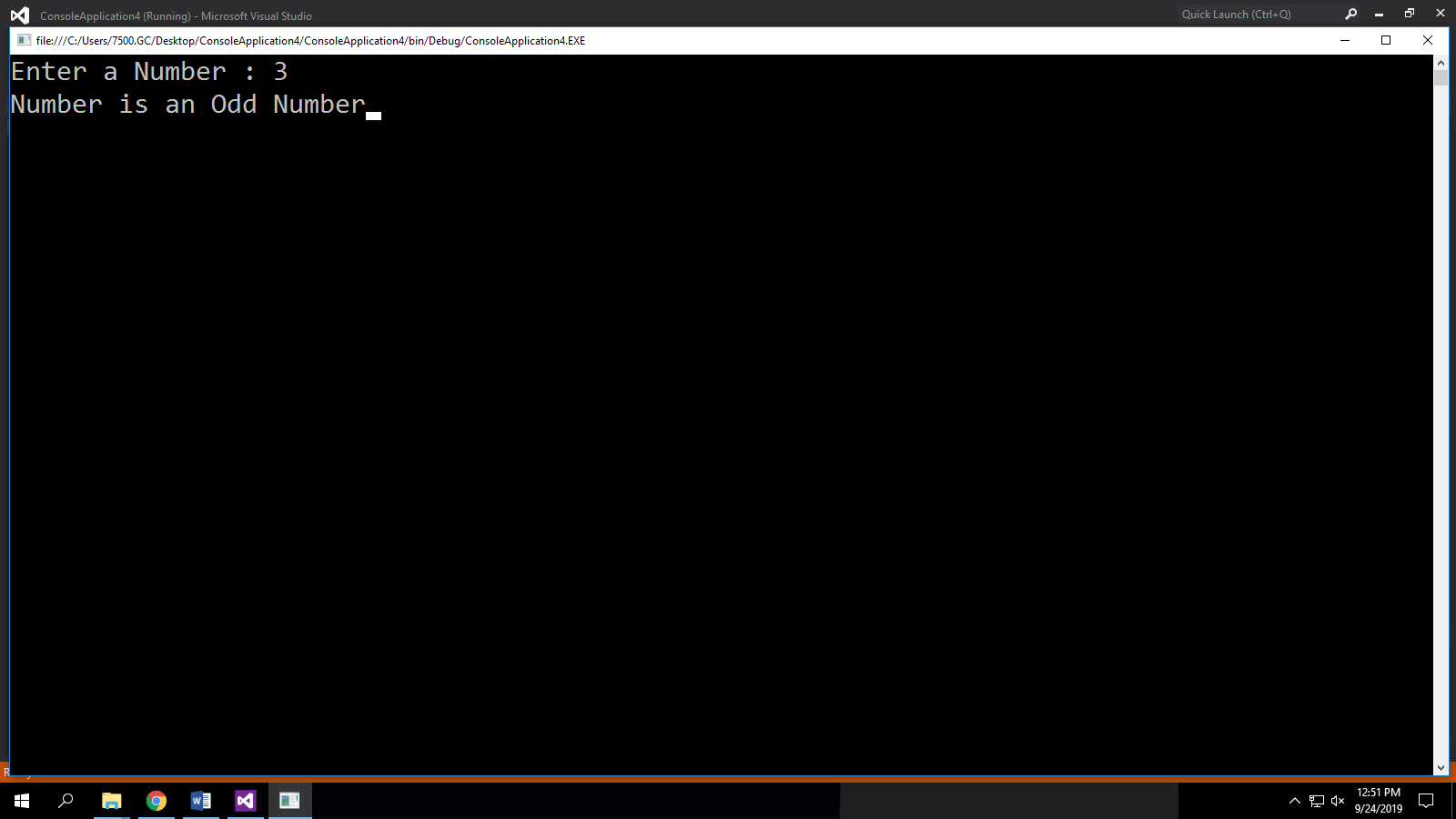
**OUTPUT**:



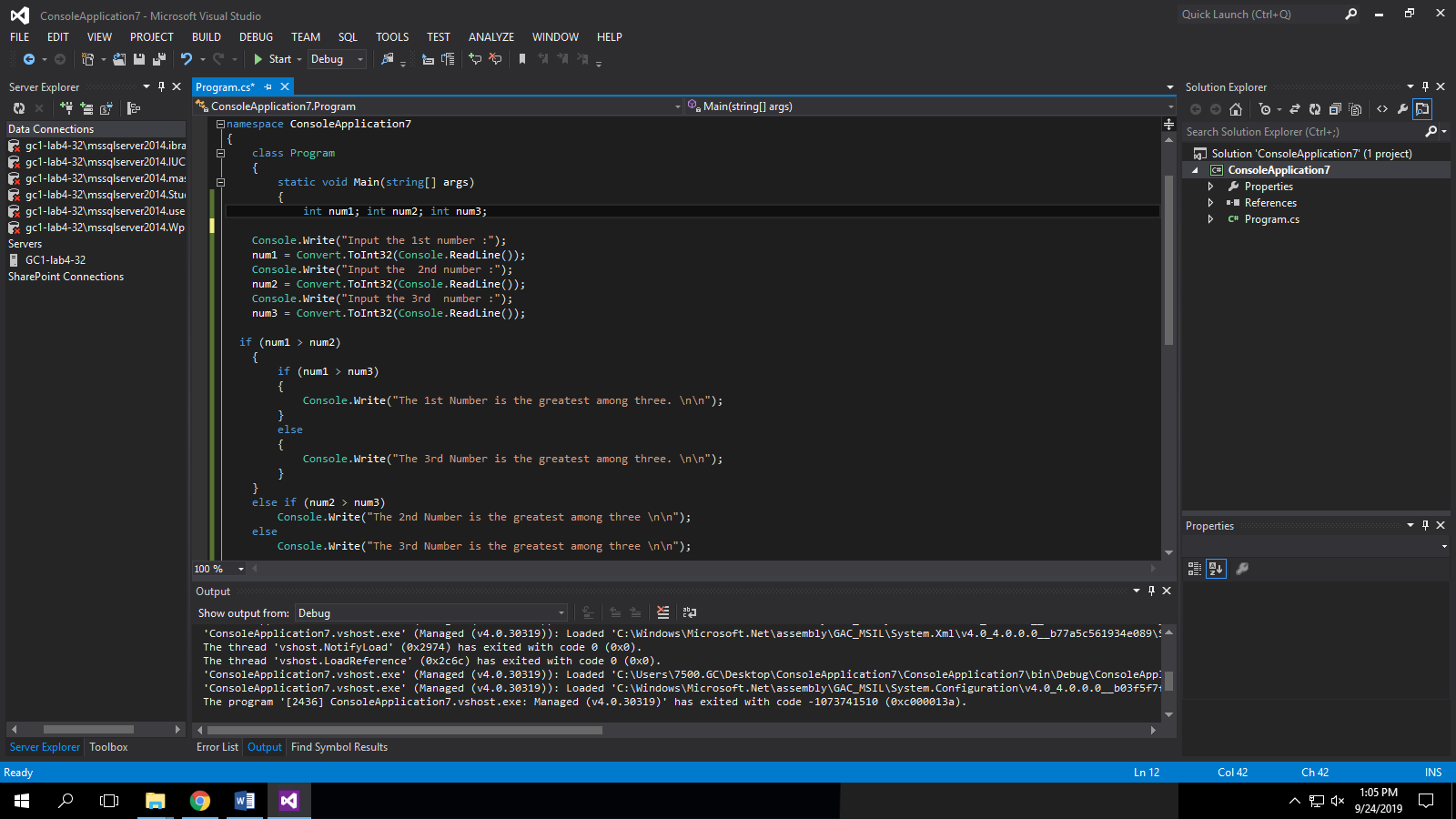
1. Write a program to calculate whether an input number is even or odd.



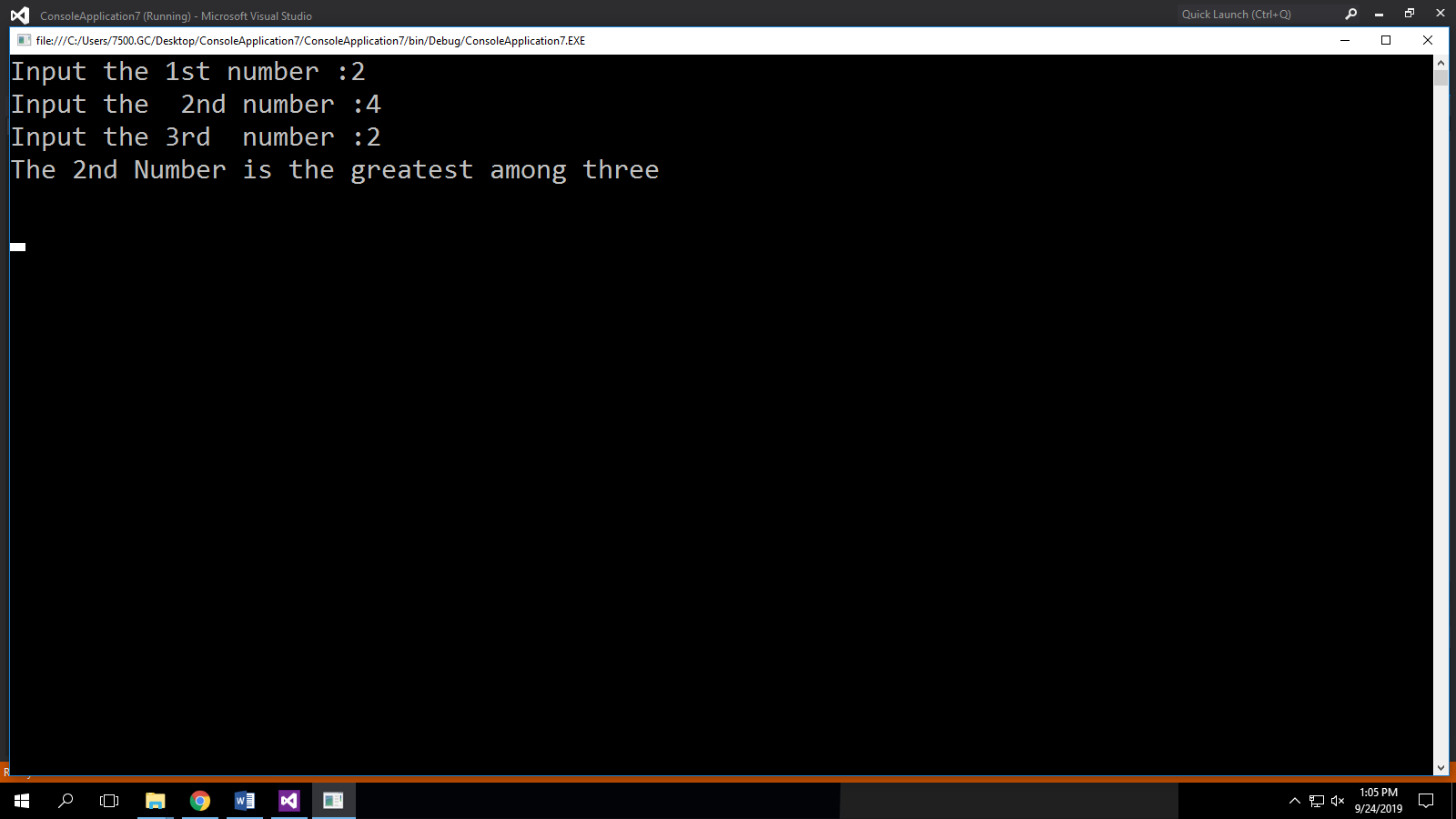
**OUTPUT:**

****

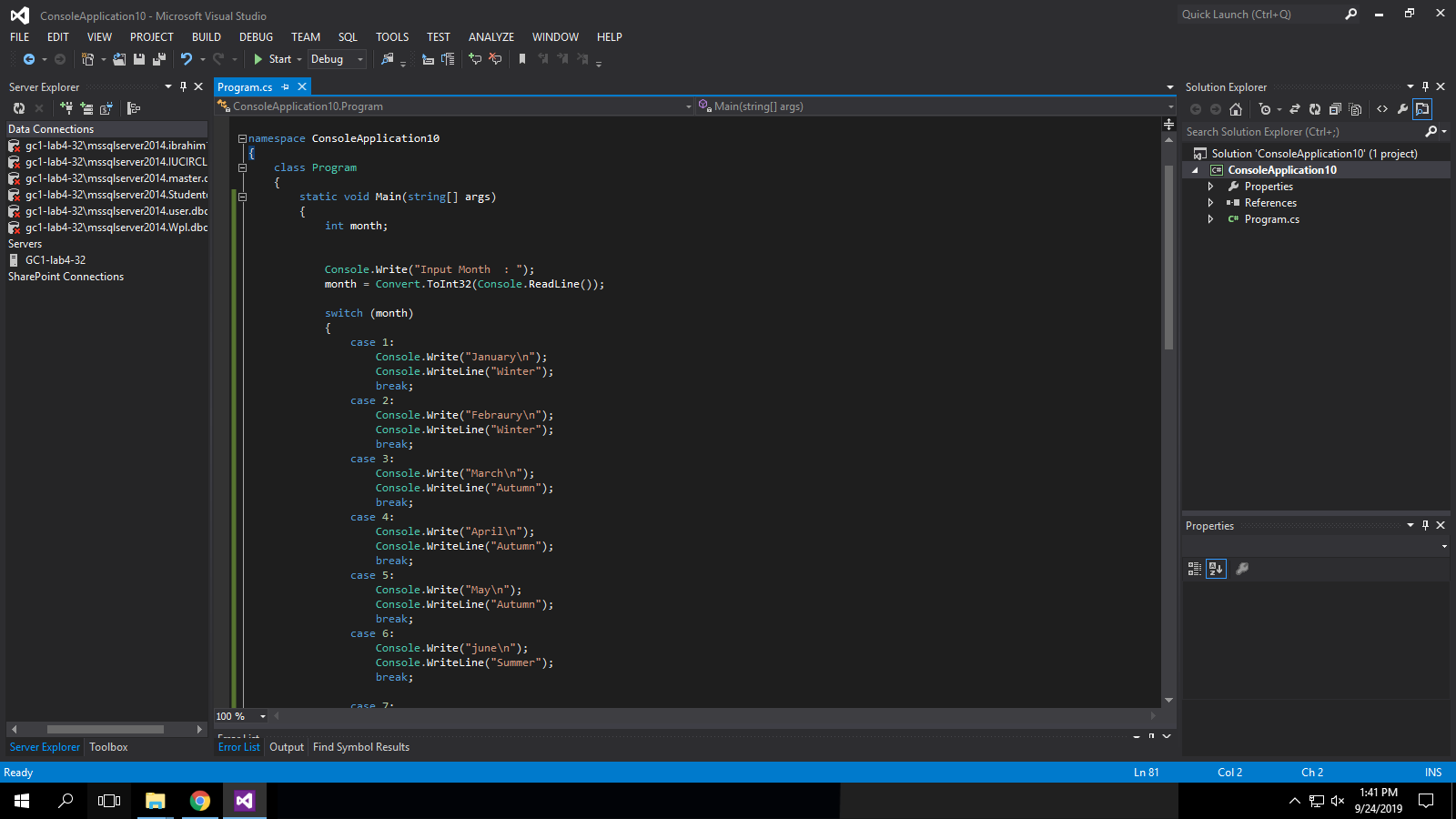
1. Write a program that takes thee numbers from user as input. The program then prints the maximum and minimum of the input numbers.

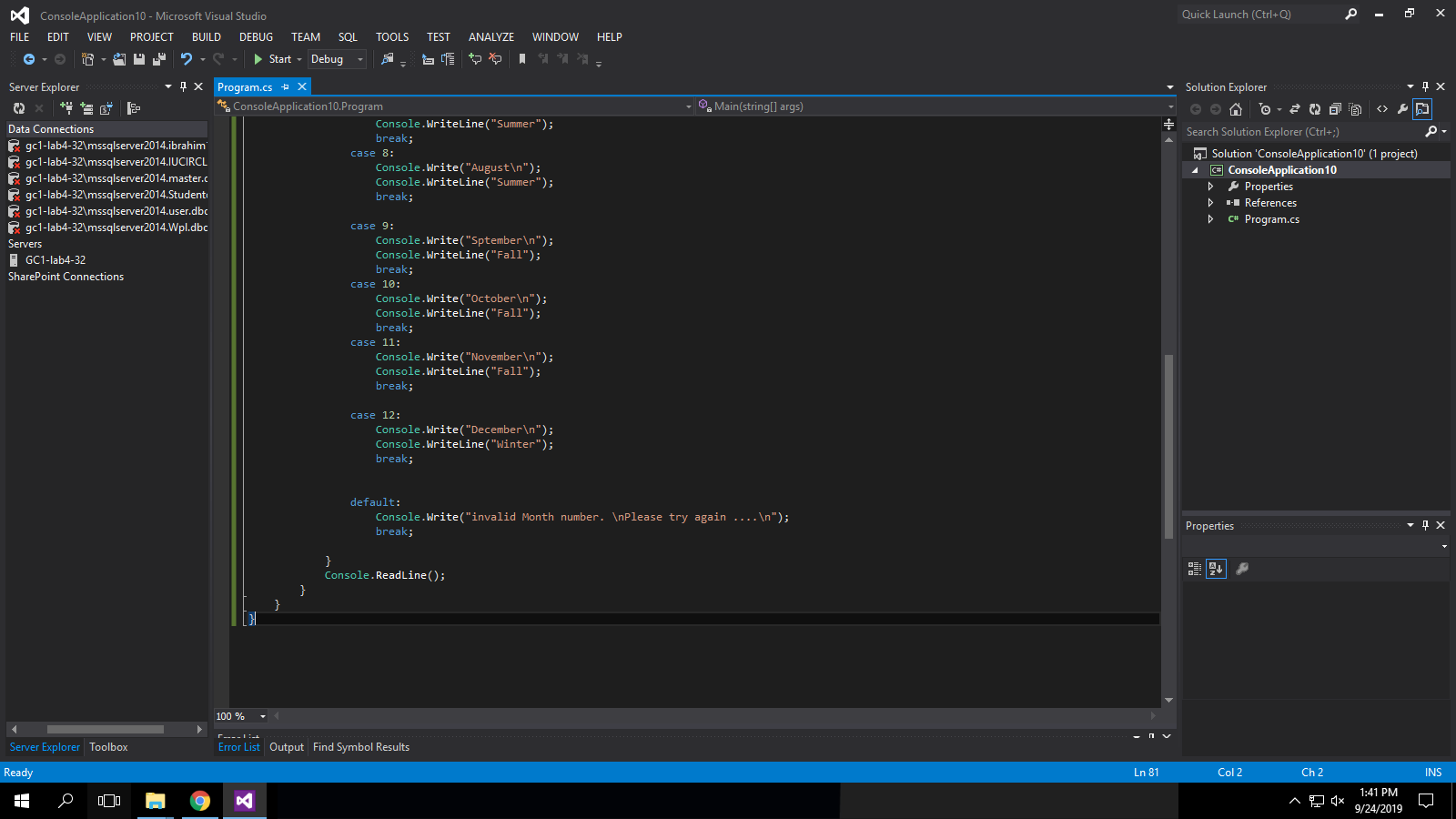


**OUTPUT:**

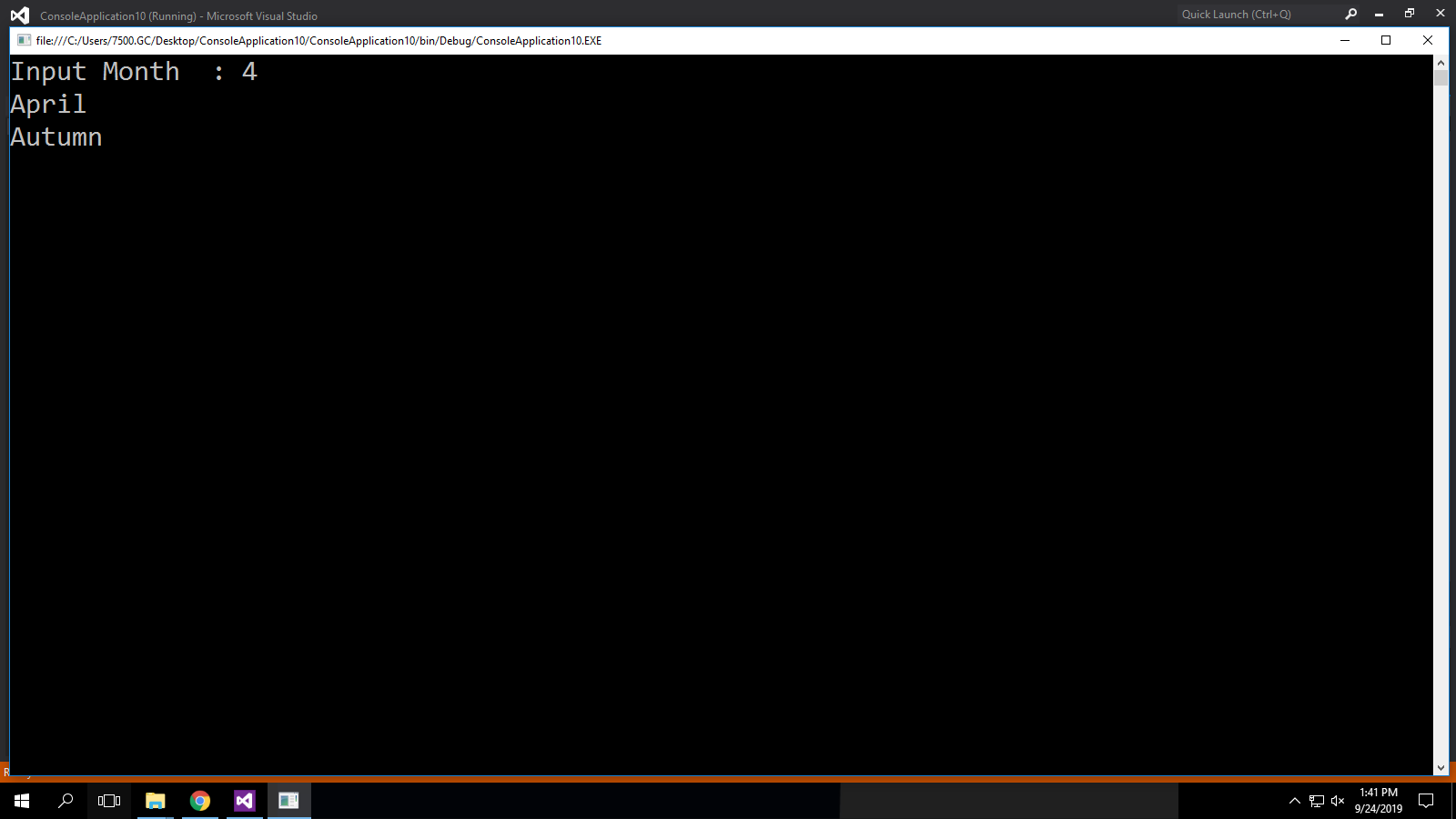
****

1. Write a program that takes the month (1…12) as input. Print whether the season is summer, winter, spring or autumn depending upon the input month.

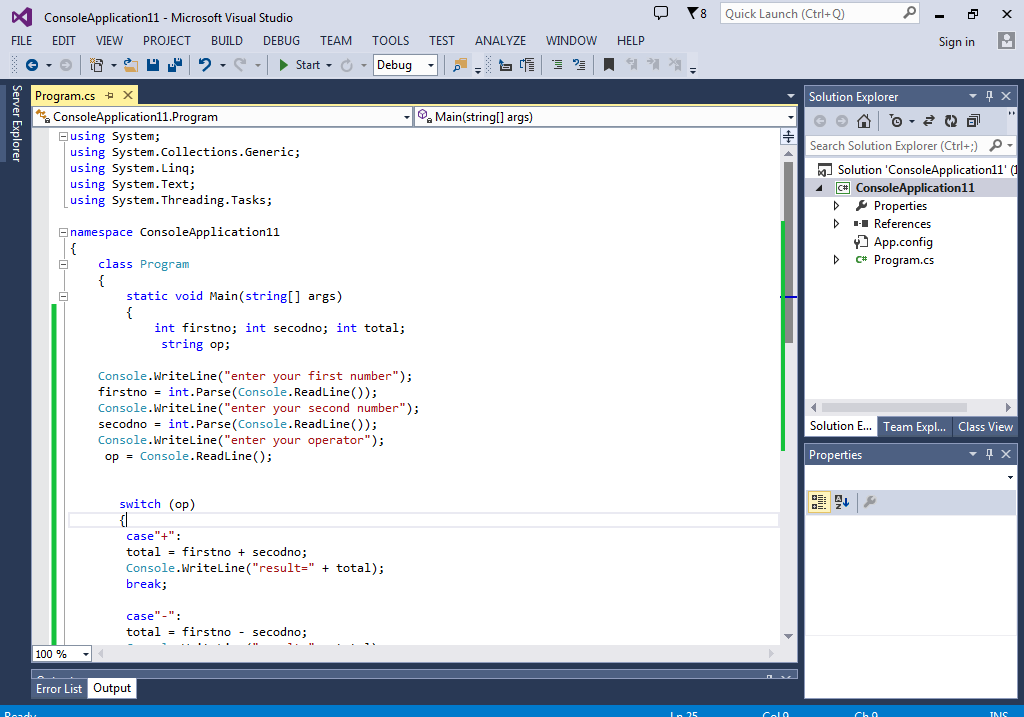


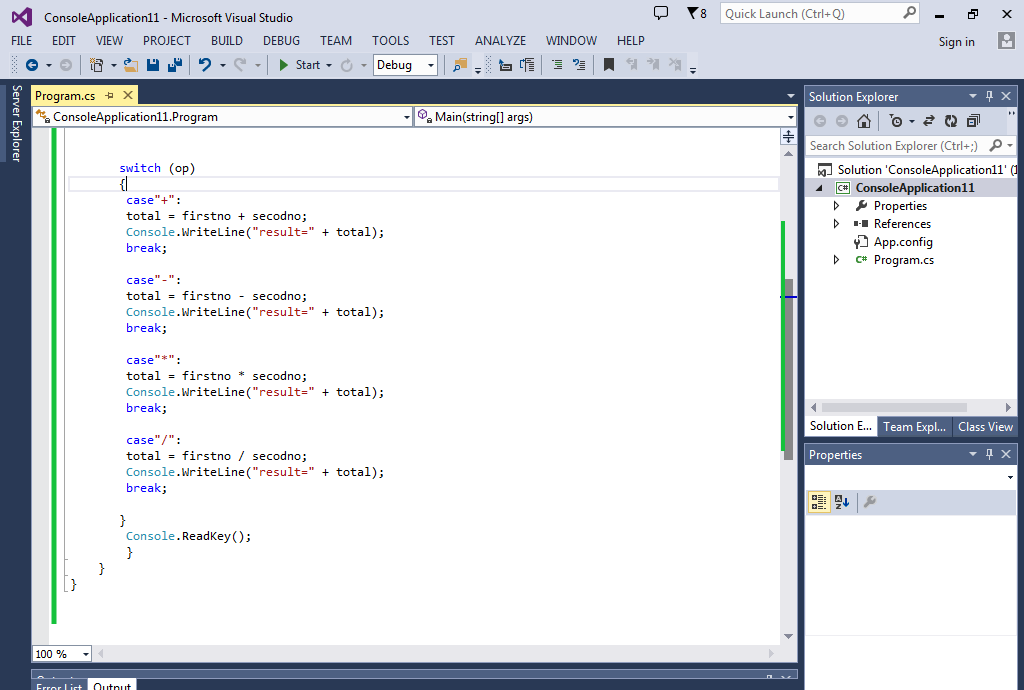


**OUTPUT:**

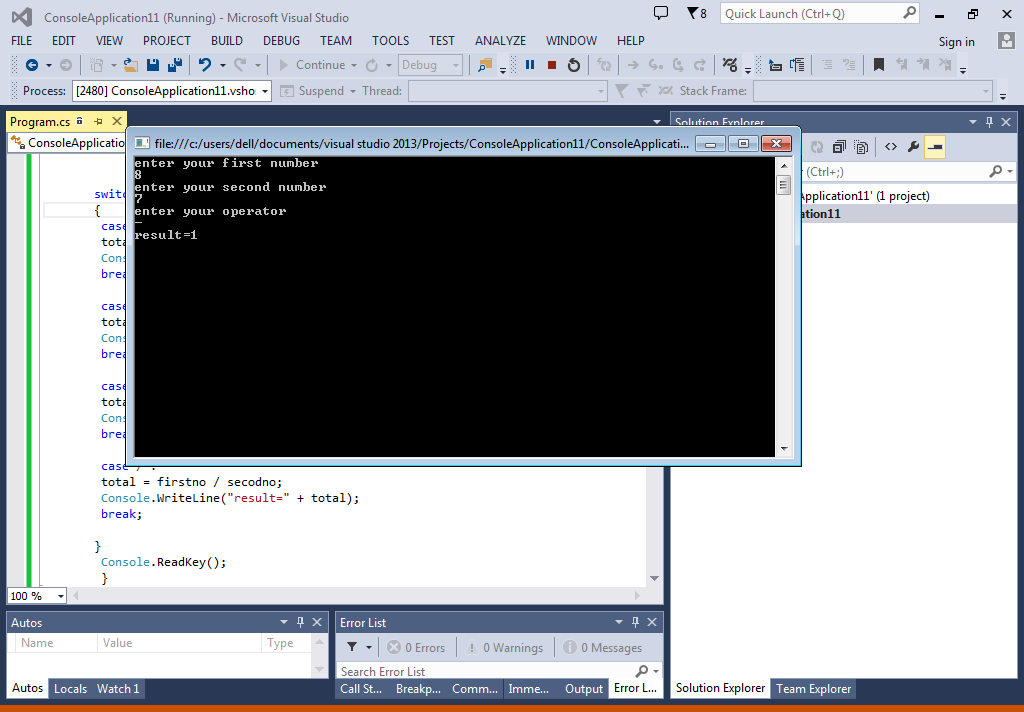


1. Write a program that takes two numbers as input and an operator as input. Using the switch statement, the program should calculate the result when the operator is applied on the two input numbers.

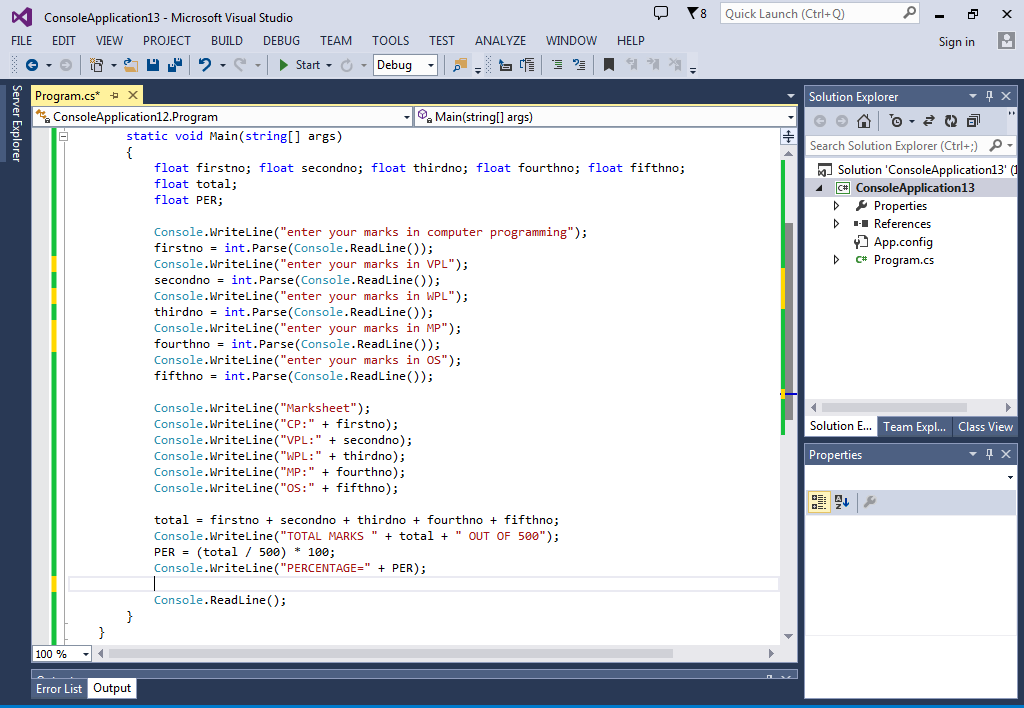




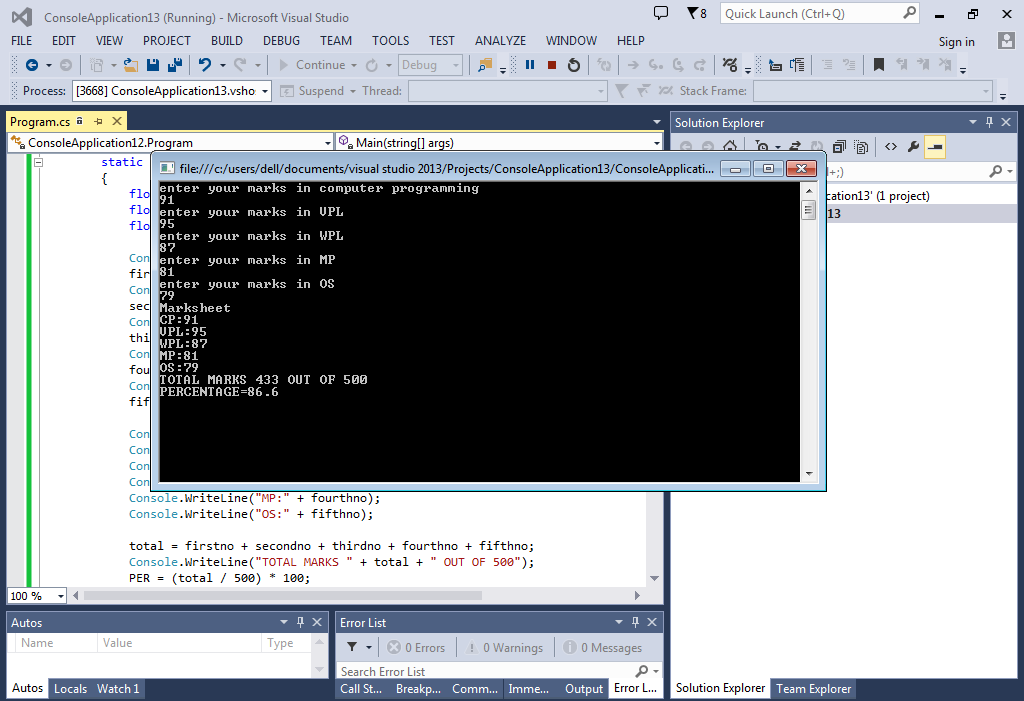
**OUTPUT:**

****

1. Write a program to print Iqra University marks sheet using if…else statement

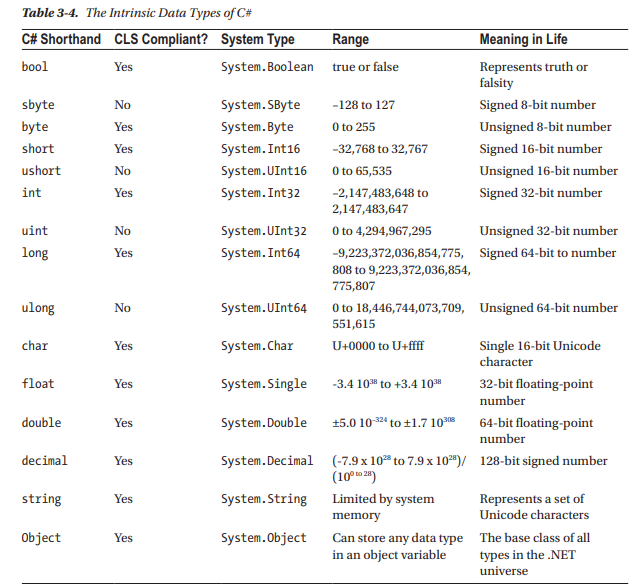


**OUTPUT:**

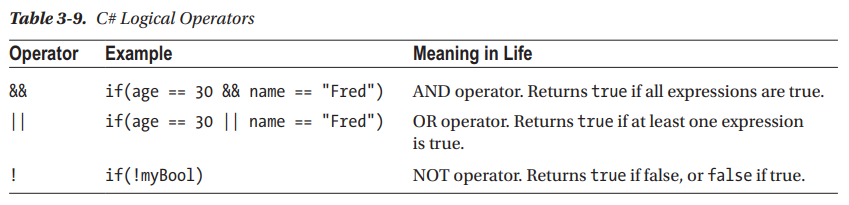
****

**Lab 2: To study and understand how to write programs in C# using loops, arrays and other constructs**

The objective of this lab is to start writing programs in C# using its basic constructs such as loops, conditions, arrays etc.Following are intrinsic data types supported by C#.



Following are the logical operators in C#:

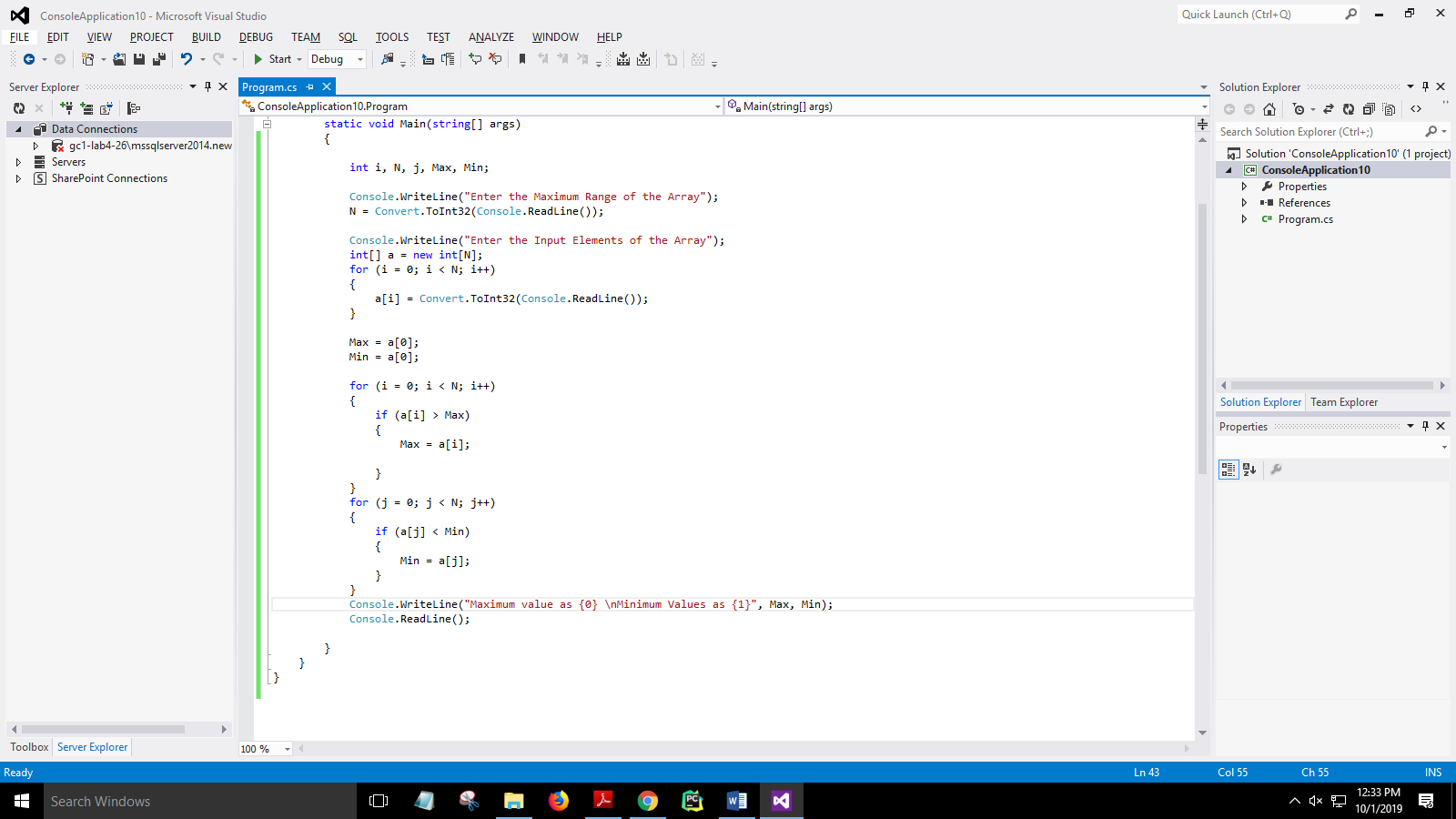


Loops are used in situations when we need to execute a block of code several number of times. C# has four types of loops: for, foreach, while and do while. An array is a collection of homogeneous data elements. You can declare an array of int as follows:

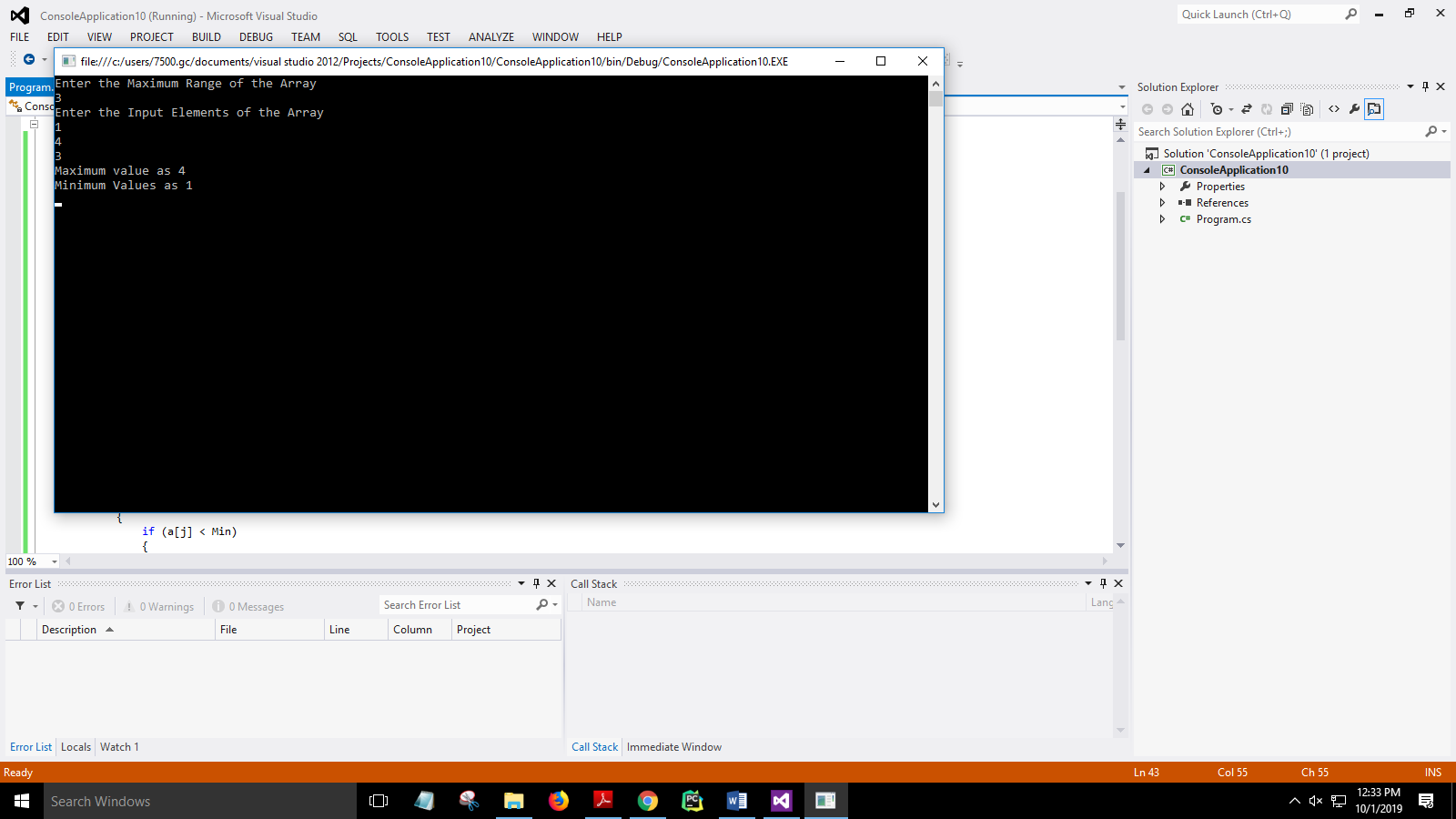
int[] myInts = new int[3];

**Lab Tasks:**

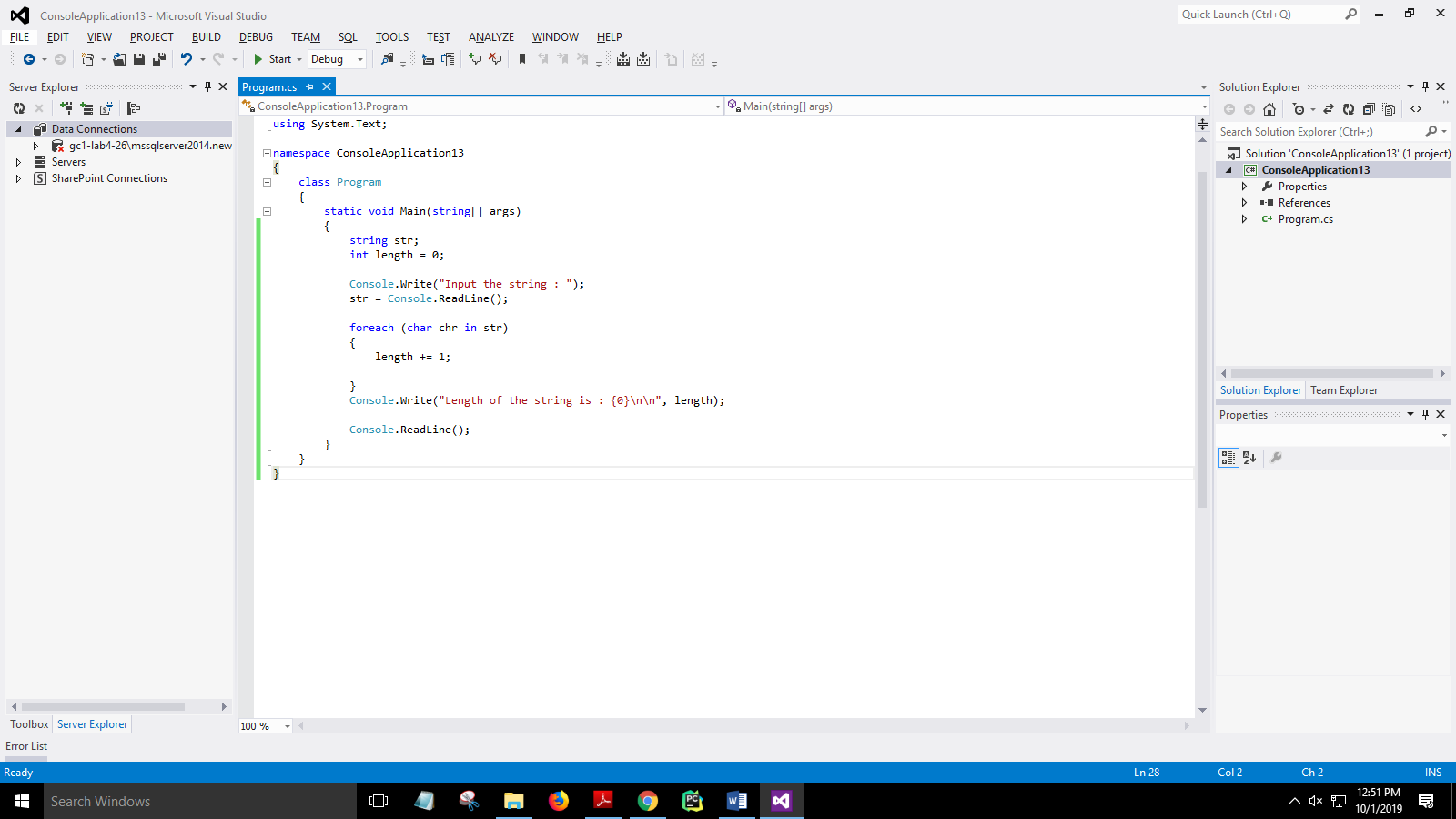
1. Write a program to find maximum and minimum element in an array.



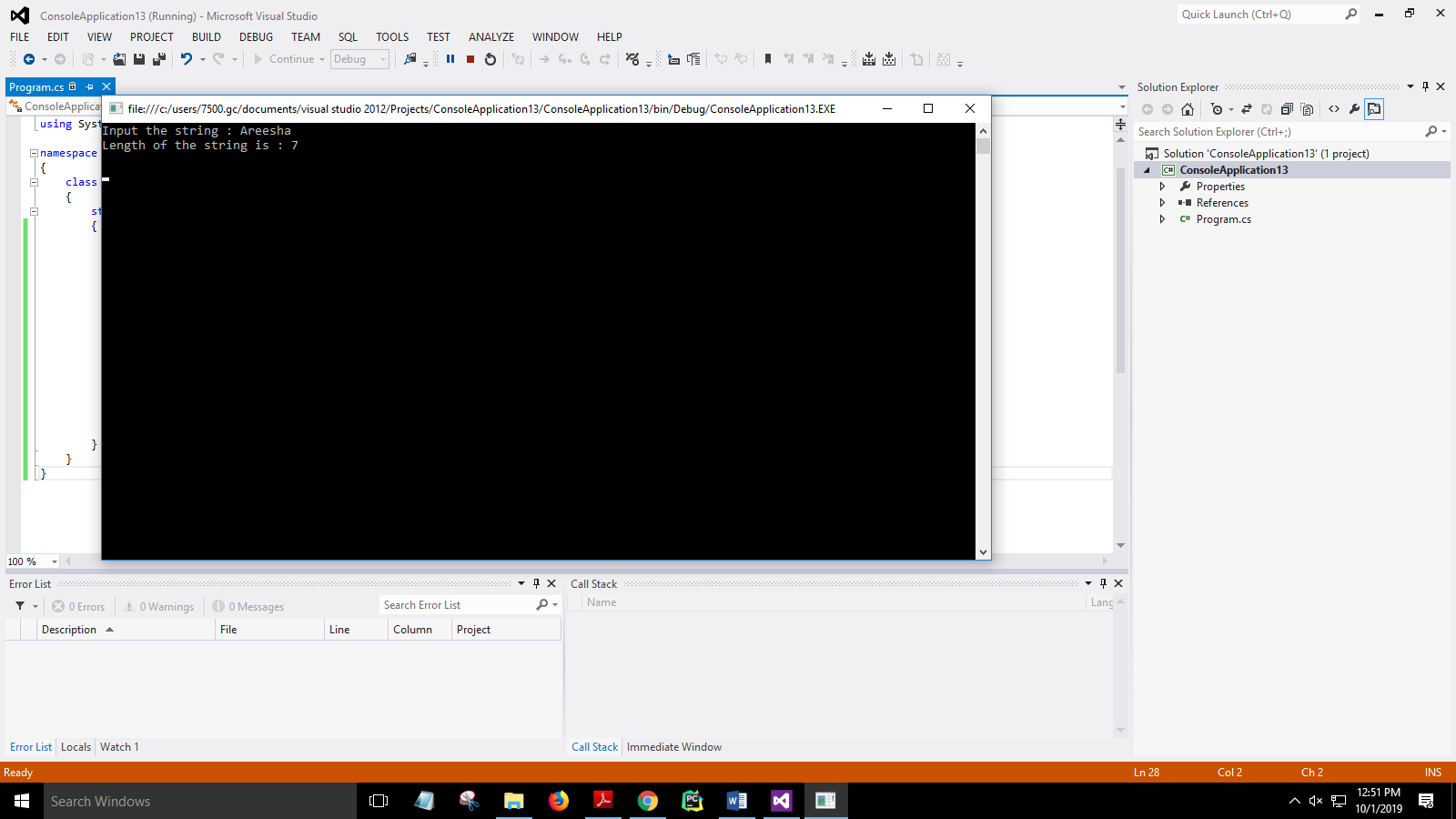
**OUTPUT:**

****

1. Write a program to find the length of a string without using library function.



**OUTPUT:**

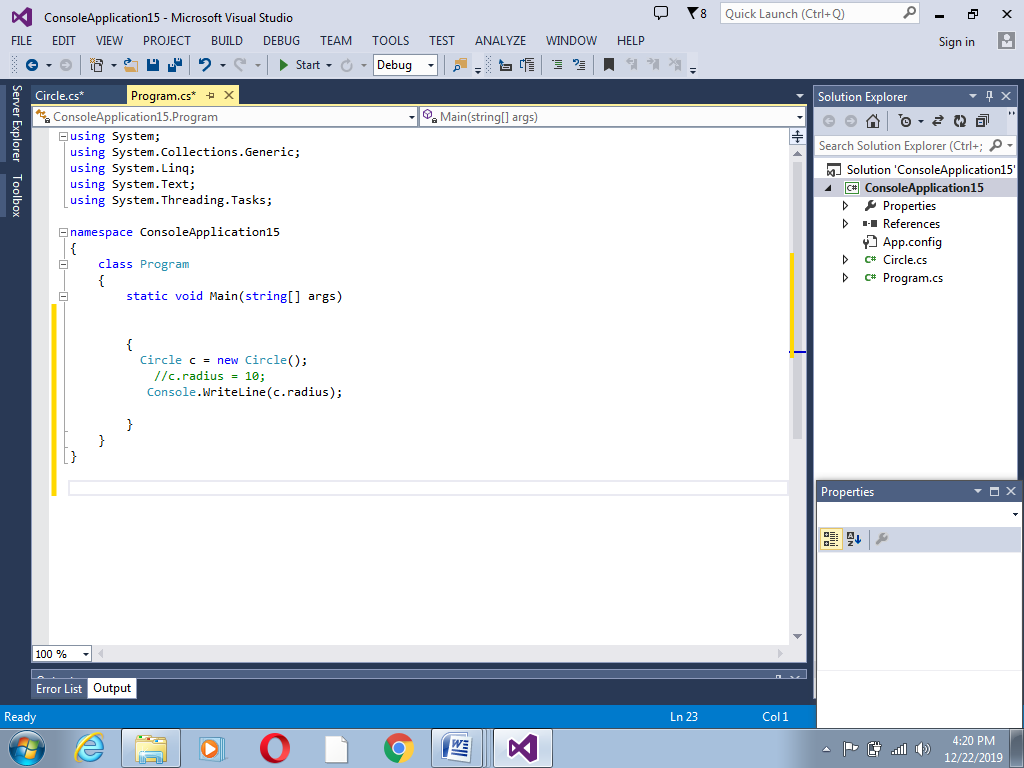
****

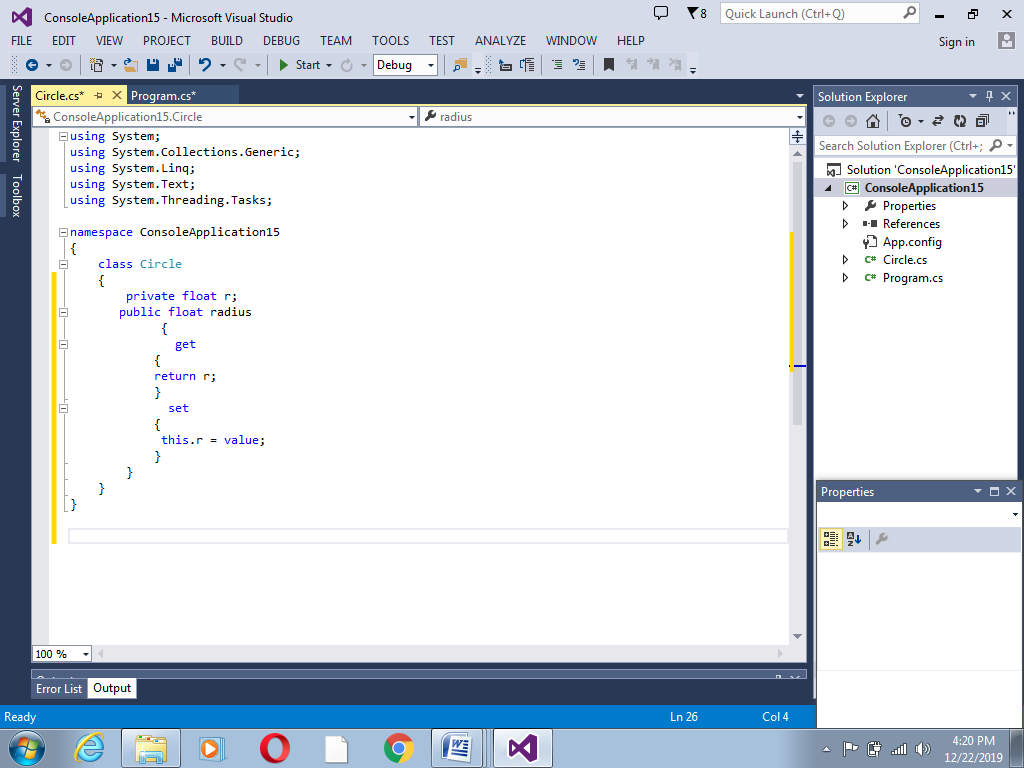
**Lab 3: To study and implement object oriented programming concepts in C#**

Object-oriented programming (OOP) refers to a type of computer programming (software design) in which programmers define not only the data type of a data structure, but also the types of operations (functions) that can be applied to the data structure. In this lab, we are going to implement the three basic pillars of OOP i.e. encapsulation, inheritance and polymorphism.

**Tasks:**

1. Encapsulation
   * + 1. Create a class Circle.
       2. Define a property named radius. In the set method, check if the radius is negative then throw an exception System.ArgumentException.
       3. Define a one argument constructor to initialize the radius.
       4. Define a no argument constructor to initialize the radius value to zero using constructor chaining.
       5. Define a method GetArea() to calculate the area of circle.
       6. Create a Test class.
       7. In the Main method, define two objects of Circle and initialize them with random values.
       8. Call the GetArea() method of each object and print the area.
2. Inheritance/Polymorphism
   * + 1. Create an abstract class Animal. Define a property: name of type string. Define an abstract method sound(). Define a constructor to initialize the animal’s name.
       2. Now define two abstract classes named Mammal and Non-Mammal that inherits the Animal class.
       3. Inherit the Mammal class to define Cat class. Implement the method sound that prints ‘Meow’.
       4. Implement classes for the Goat inheriting the Mammal and the Fish inheriting the Non-Mammal class
       5. Create a Test class. Define few objects of classes Cat, Goat and Fish. Assign the instance variables to reference variable of Animal class and polymorphically call them.





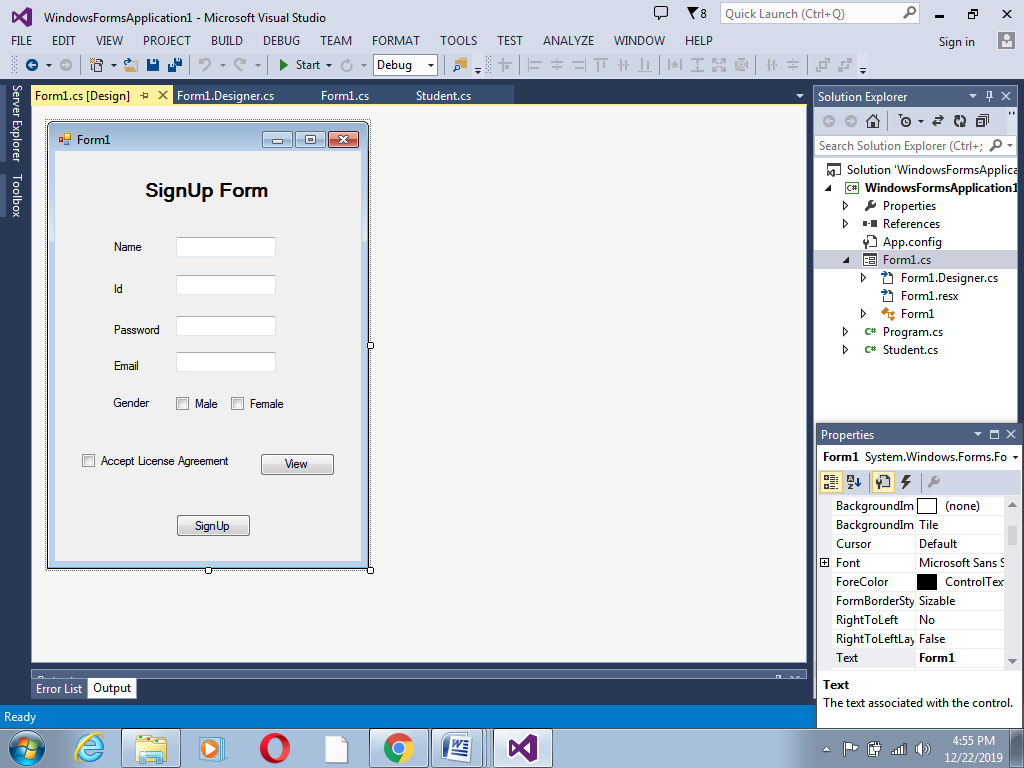
**Lab 4: To study and implement Windows Forms application in C#**

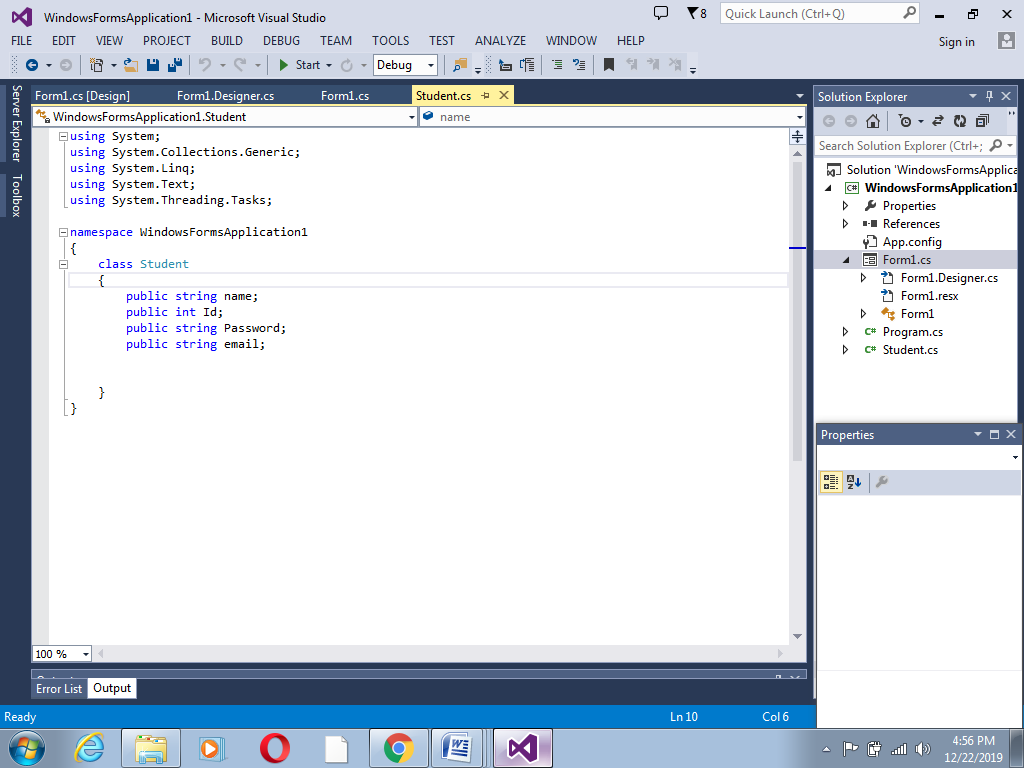
In this lab we will implement windows forms in C#. Windows Forms (WinForms) is a graphical (GUI) class library included as a part of Microsoft .NET Framework or Mono Framework, providing a platform to write rich client applications for desktop, laptop, and tablet PCs.

We will look at some of the most common controls used in C# and use them to develop a basic application.

**Tasks:**

1. Design a basic Sign Up form with fields name, id, password, email, address, gender and date of birth.
2. Add a checkbox to accept license agreement, and a view button to view the license agreement.
3. Add a Sign Up button.
4. When the user clicks on Sign Up, following validations are to be performed:
   1. Password must contain a capital letter and one digit
   2. Email address should be in proper format
   3. All the fields must be filled by user
   4. The user must be 18 years or above
   5. The user has accepted the license agreement
5. When the user clicks on view license agreement button, a new form is to be displayed with the license agreement.



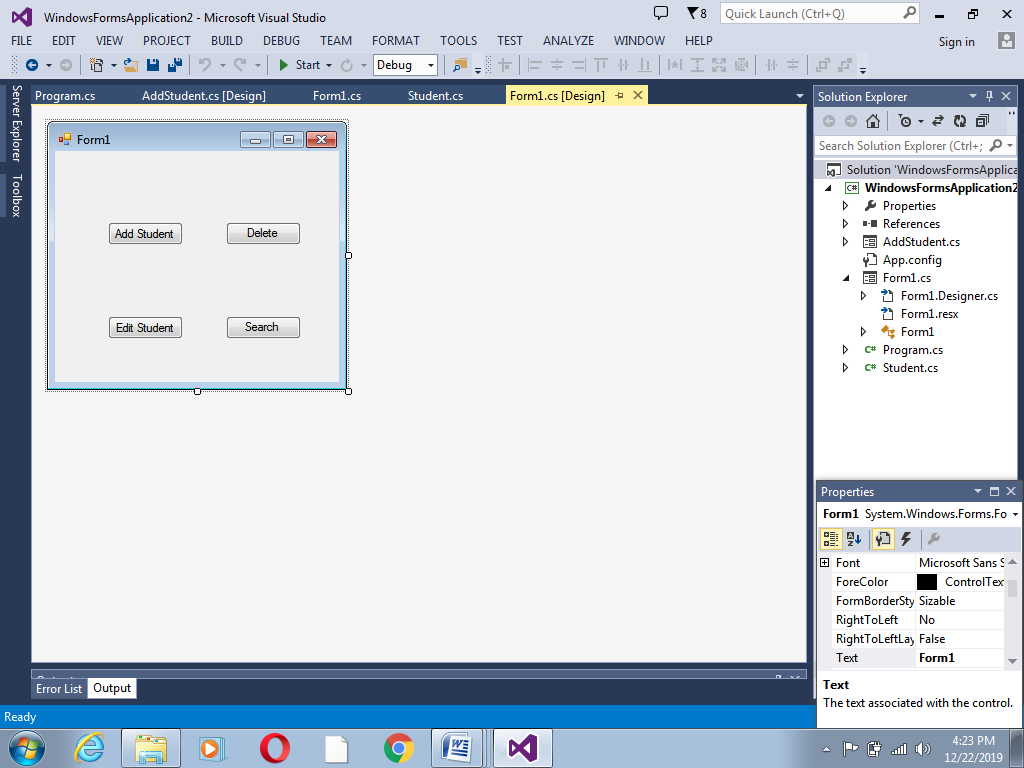


**Lab 5: To study and implement Collections in C#**

In todays, lab we will implement the collections in C#. A collection is an abstract data type for grouping together multiple values. It's therefore sometime known as container. A collection is just a grouping of some objects with the same type.

**Lab Tasks:**

1. Create a WinForm in C# with following buttons: Add a Student, Edit a Student, Delete a Student, Search a Student.
2. Now, create a class Student with the following properties: name, age, gender, cgpa
3. Implement the Add a Student button. When the user clicks this button, open a new form from which user can provide the details. Save the results in a collection.
4. Implement the Edit button such that user can modify the students record. The changes are reflected in the collection.
5. Implement the Delete and Search options.



//student class

using System;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

usingSystem.Windows.Forms;

namespace WindowsFormsApplication2

{

publicpartialclassAddStudent :Form

{

privateList<Student> students;

privateStudent s;

publicAddStudent(List<Student> students)

{

this.students = students;

InitializeComponent();

}

publicAddStudent(Student s)

{

this.s = s;

InitializeComponent();

this.loadData();

this.adjustControls();

}

publicvoidadjustControls()

{

this.Text = "Edit Student Data";

this.button2.Text = "Edit";

this.sname.Enabled = false;

}

publicvoidloadData()

{

this.sname.Text = s.name;

this.age.Text = s.age.ToString();

this.cgpa.Text = s.cgpa.ToString();

if (s.gender.Equals("Male"))

{

this.gender.SelectedIndex = 0;

}

else

{

this.gender.SelectedIndex = 1;

}

}

privatevoid button1\_Click(object sender, EventArgs e)

{

this.Close();

}

privatevoid button2\_Click(object sender, EventArgs e)

{

if (button2.Text == "Add")

{

Student s = newStudent();

s.name = this.sname.Text;

s.age = int.Parse(this.age.Text);

s.cgpa = float.Parse(this.cgpa.Text);

s.gender = this.gender.SelectedItem.ToString();

students.Add(s);

MessageBox.Show("Student added successfully");

}

else

{

s.name = this.sname.Text;

s.age = int.Parse(this.age.Text);

s.cgpa = float.Parse(this.cgpa.Text);

s.gender = this.gender.SelectedItem.ToString();

MessageBox.Show("Student edited successfully");

}

this.Close();

}

privatevoidAddStudent\_Load(object sender, EventArgs e)

{

}

}

}

using System;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

usingSystem.Windows.Forms;

namespace WindowsFormsApplication2

{

publicpartialclassListStudent :Form

{

List<Student> students;

publicListStudent(List<Student> students)

{

this.students = students;

InitializeComponent();

}

privatevoidListStudent\_Load(object sender, EventArgs e)

{

this.student\_list.DataSource = this.students;

}

publicvoidselect\_edit()

{

this.Edit.Checked = true;

}

publicvoidselect\_delete()

{

this.Delete.Checked = true;

}

privatevoid button1\_Click(object sender, EventArgs e)

{

string name = this.student\_list.SelectedItem.ToString();

Student s = newStudent();

s.name = name;

inti = students.IndexOf(s);

if (Delete.Checked)

{

students.RemoveAt(i);

this.student\_list.DataSource = null;

this.student\_list.DataSource = students;

}

else

{

s = students[i];

AddStudent add = newAddStudent(s);

add.Show();

}

}

}

}

using System;

usingSystem.Collections;

usingSystem.Collections.Generic;

usingSystem.ComponentModel;

usingSystem.Data;

usingSystem.Drawing;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

usingSystem.Windows.Forms;

namespace WindowsFormsApplication2

{

publicpartialclassForm1 :Form

{

privateList<Student> students = newList<Student>();

public Form1()

{

InitializeComponent();

}

privatevoid button1\_Click(object sender, EventArgs e)

{

AddStudent add = newAddStudent(students);

add.Show();

}

privatevoid Form1\_Load(object sender, EventArgs e)

{

Student s1 = newStudent();

s1.name = "Ali";

s1.gender = "Male";

s1.age = 21;

s1.cgpa = 3.6f;

students.Add(s1);

s1 = newStudent();

s1.name = "Areeba";

s1.gender = "Female";

s1.age = 20;

s1.cgpa = 3.2f;

students.Add(s1);

}

privatevoid button2\_Click(object sender, EventArgs e)

{

ListStudent ls = newListStudent(this.students);

ls.select\_edit();

ls.Show();

}

privatevoid button4\_Click(object sender, EventArgs e)

{

ListStudent ls = newListStudent(this.students);

ls.select\_delete();

ls.Show();

}

}

}

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

namespace WindowsFormsApplication2

{

publicclassStudent

{

publicstring name { get; set; }

publicint age { get; set; }

publicfloatcgpa{ get; set; }

publicstring gender { get; set; }

publicoverrideStringToString()

{

return name;

}

publicoverrideboolEquals(object obj)

{

if(!Convert.IsDBNull(obj))

{

Student s = (Student)obj;

returnthis.name.Equals(s.name);

}

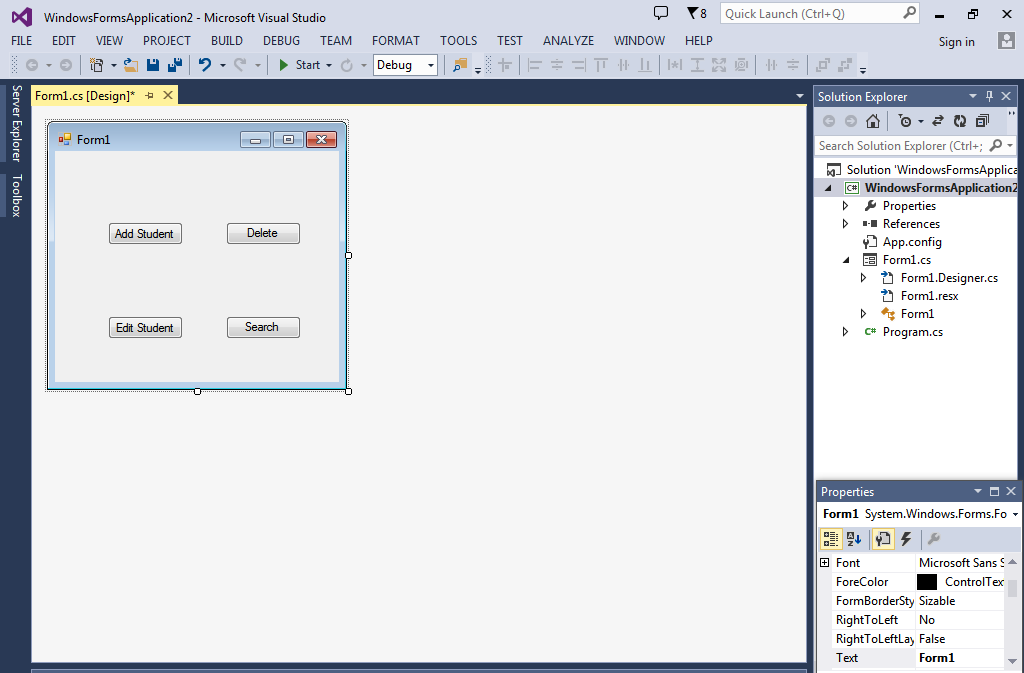
returnfalse;

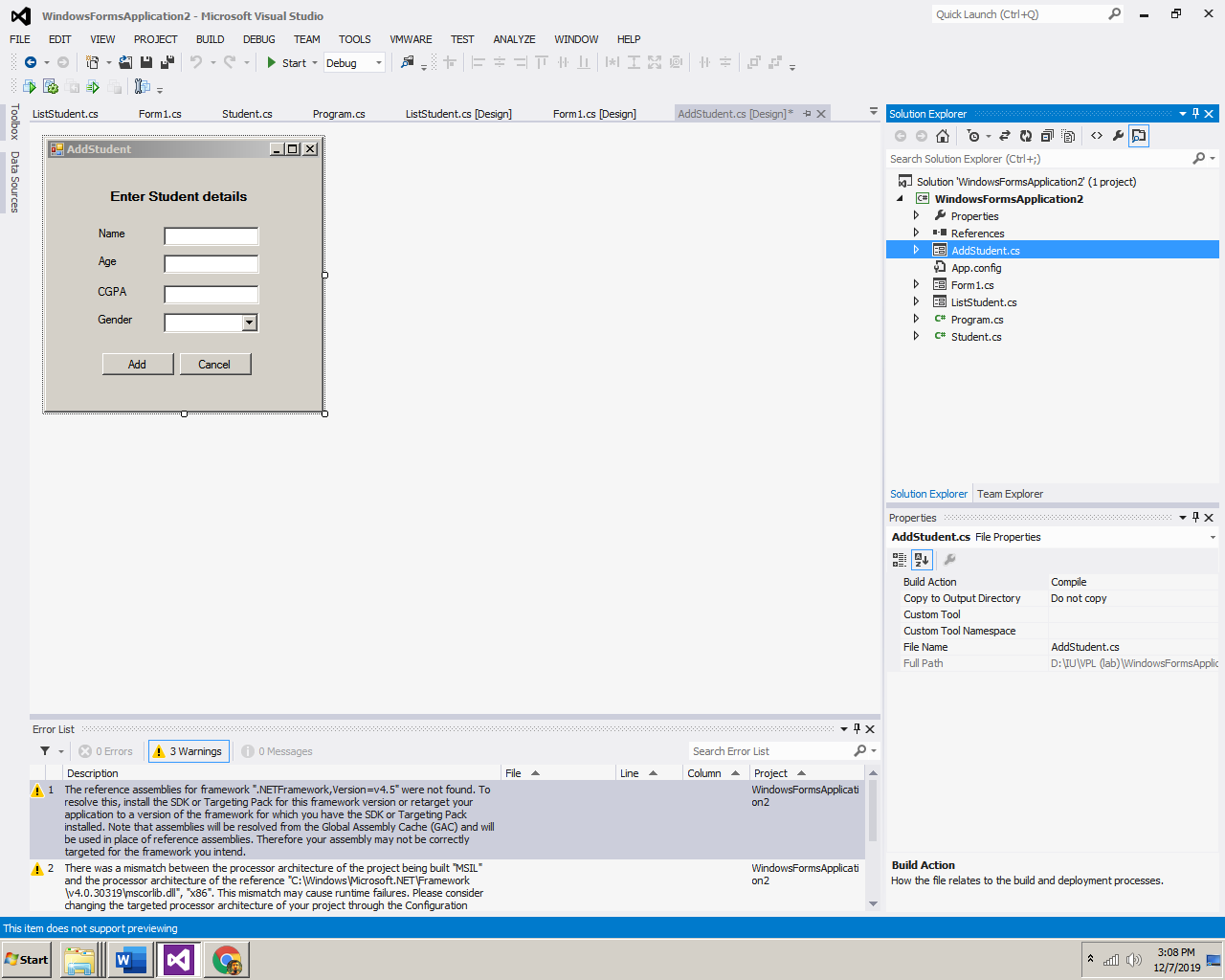
}

}

}

**OUTPUT:**

****

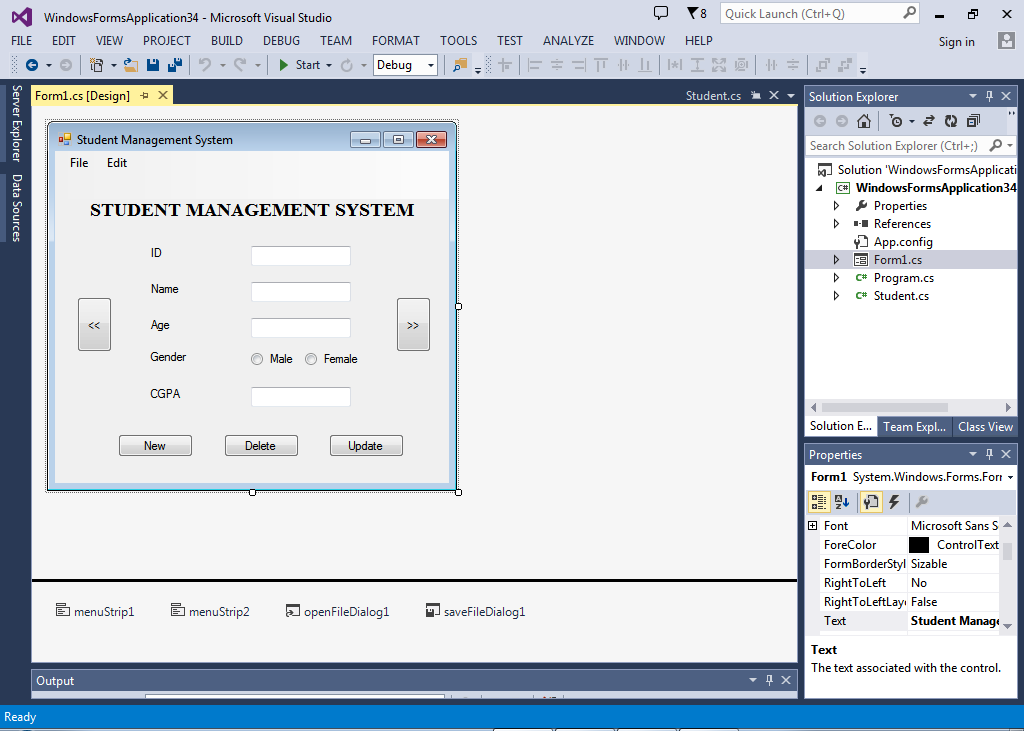
****

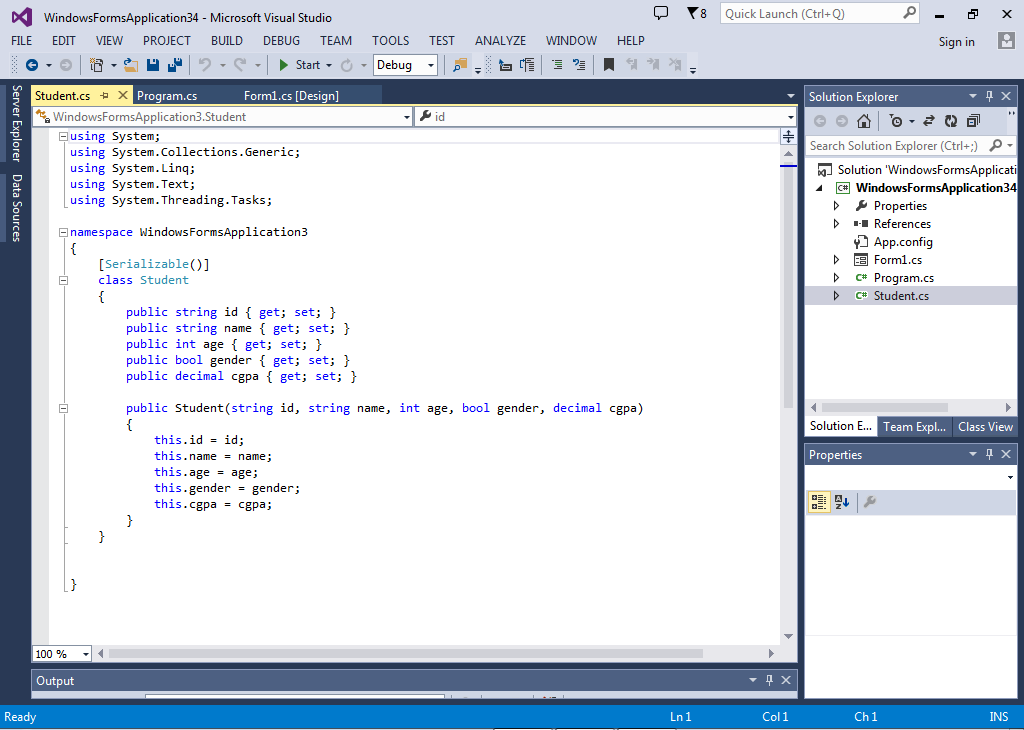
**Lab 6: To study and implement I/O in C#**

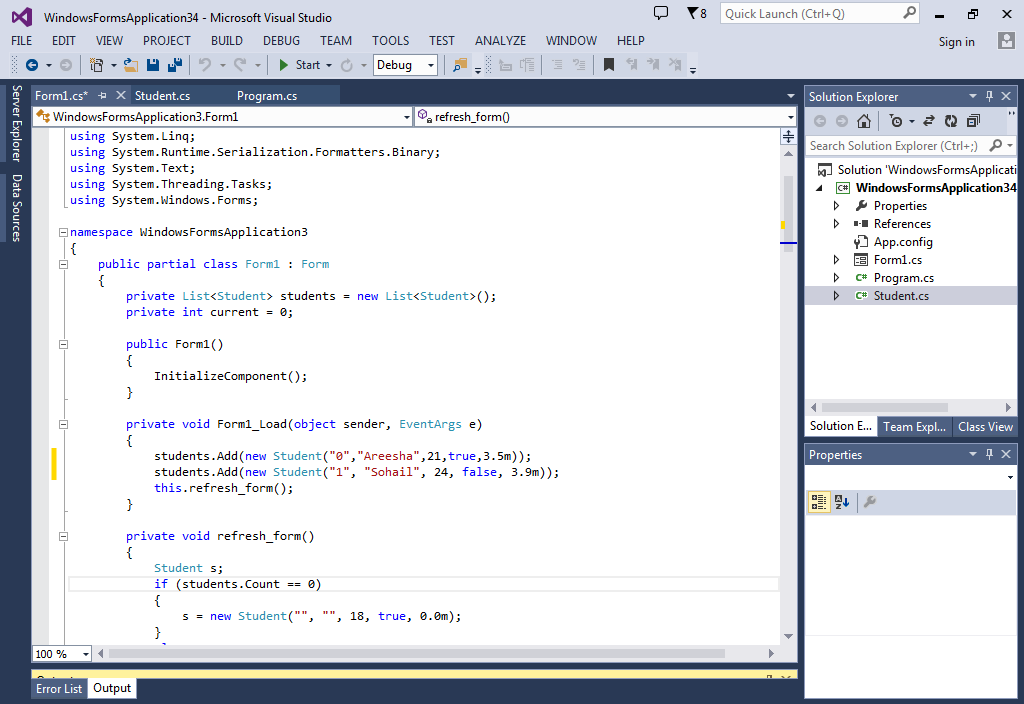
In this lab, we will use StreamReader and StreamWriter of C# to develop a basic application that can insert student’s records in a file. A stream can be defined as a sequence of data. The StreamReader is used to read data from a source and the StreamWriter is used for writing data to a destination.

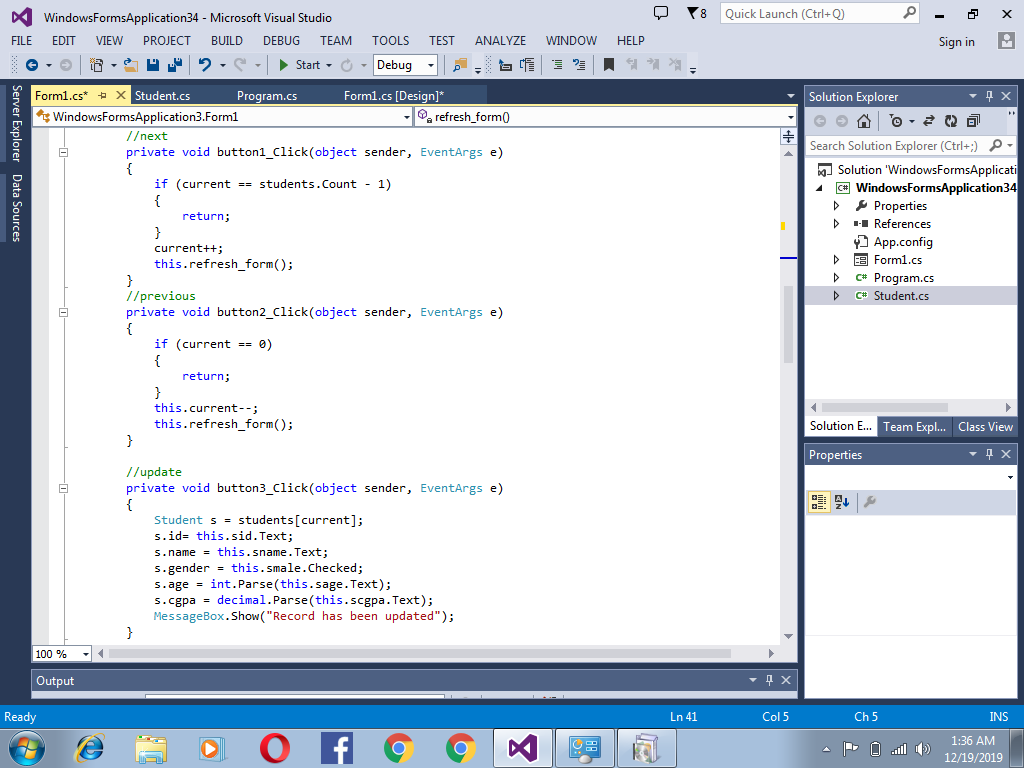
**Lab Tasks:**

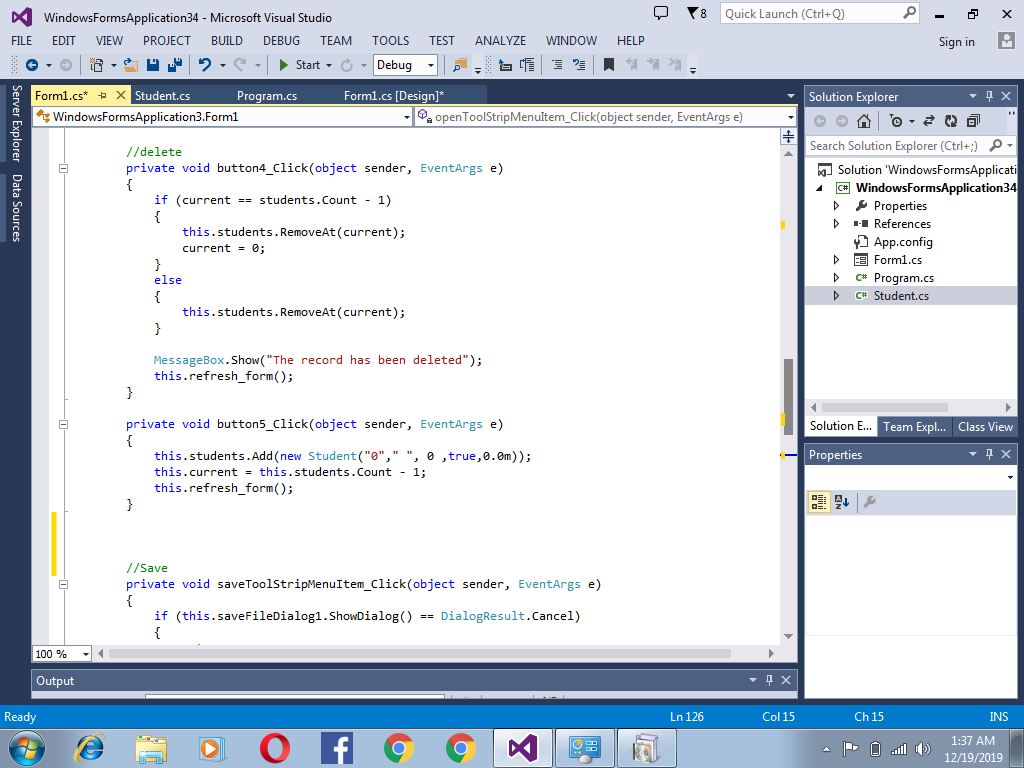
1. Create a basic form in C# with the following fields: student id, age, gender, marks.
2. Add the buttons to add a new record, delete a record and update a record, in memory.
3. Create a Menu Strip from which user can save the records in a file or load the records from a file.

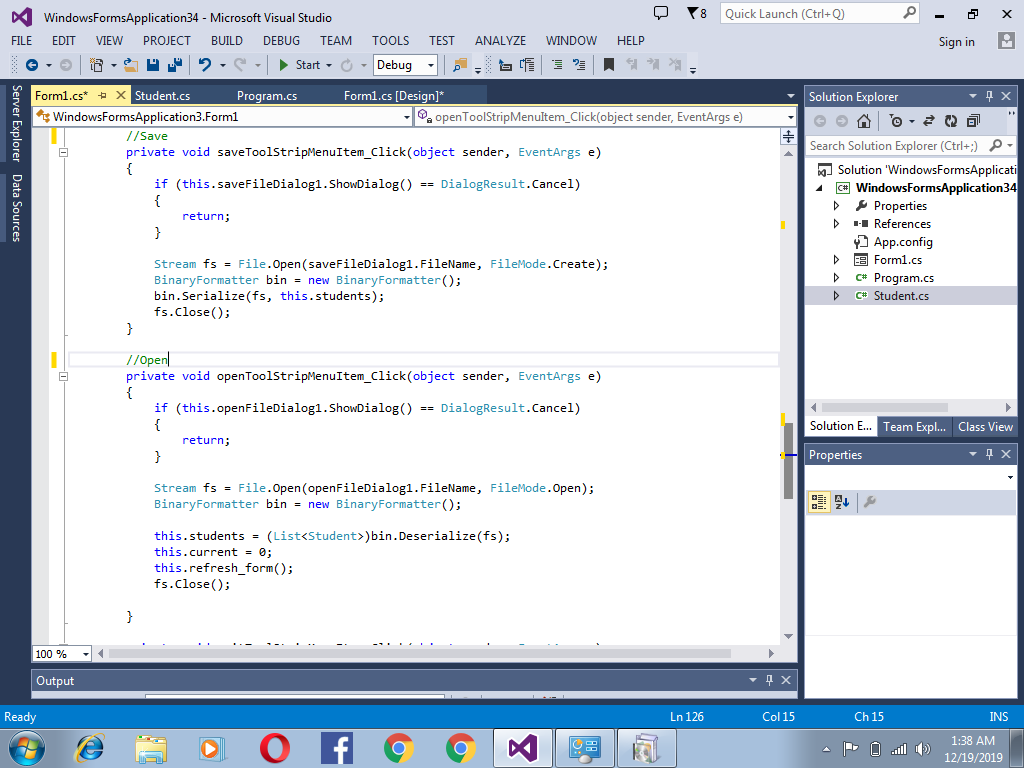


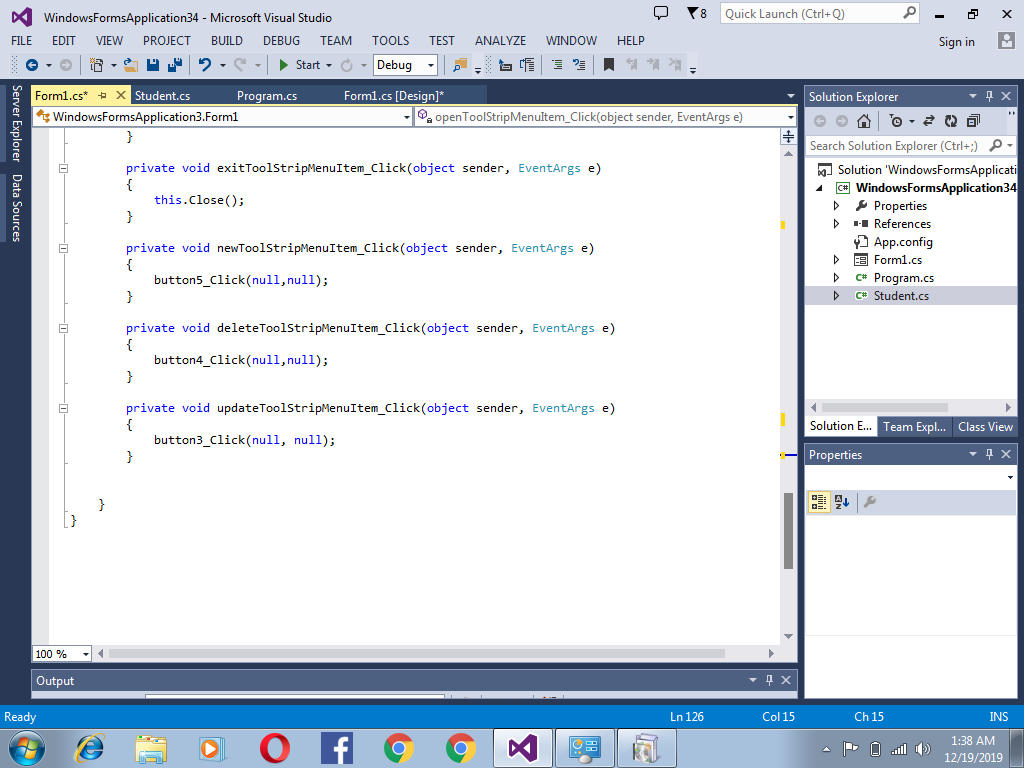




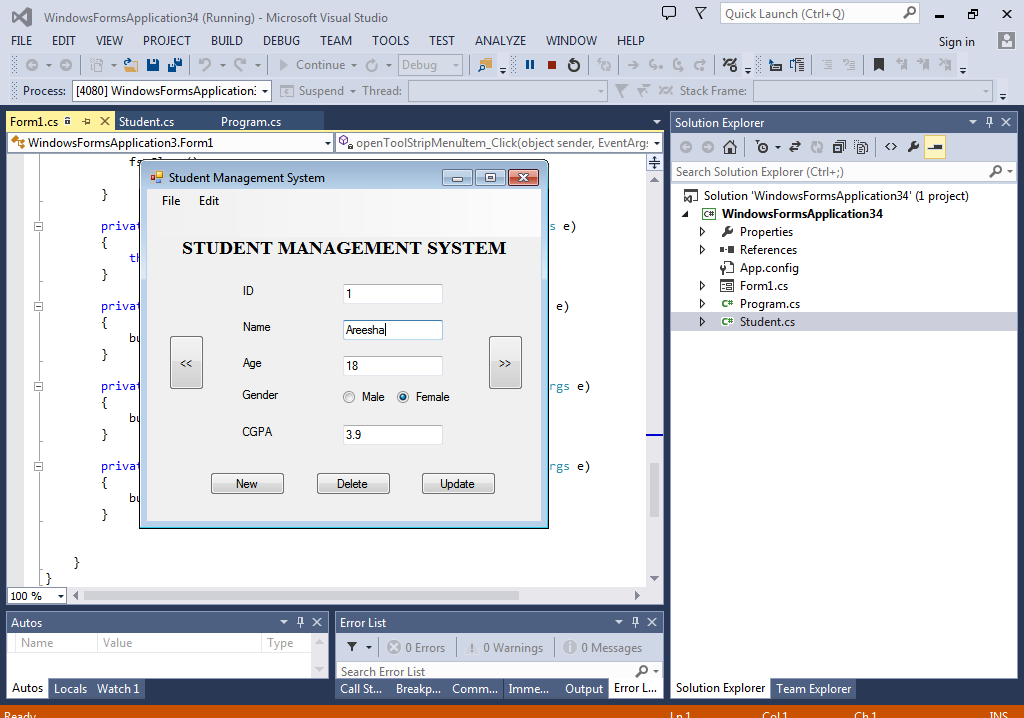








**OUTPUT:**

****

**Lab 7:To study and implement XML parsing in C#**

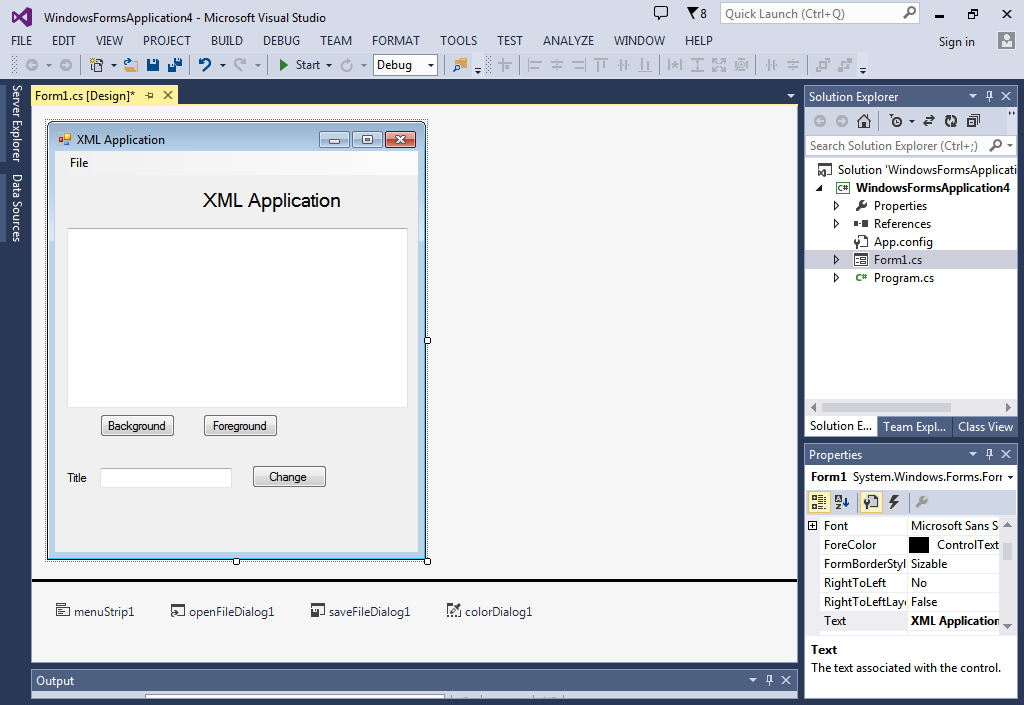
Extensible Markup Language (XML) defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. It is a software- and hardware-independent tool for storing and transporting data

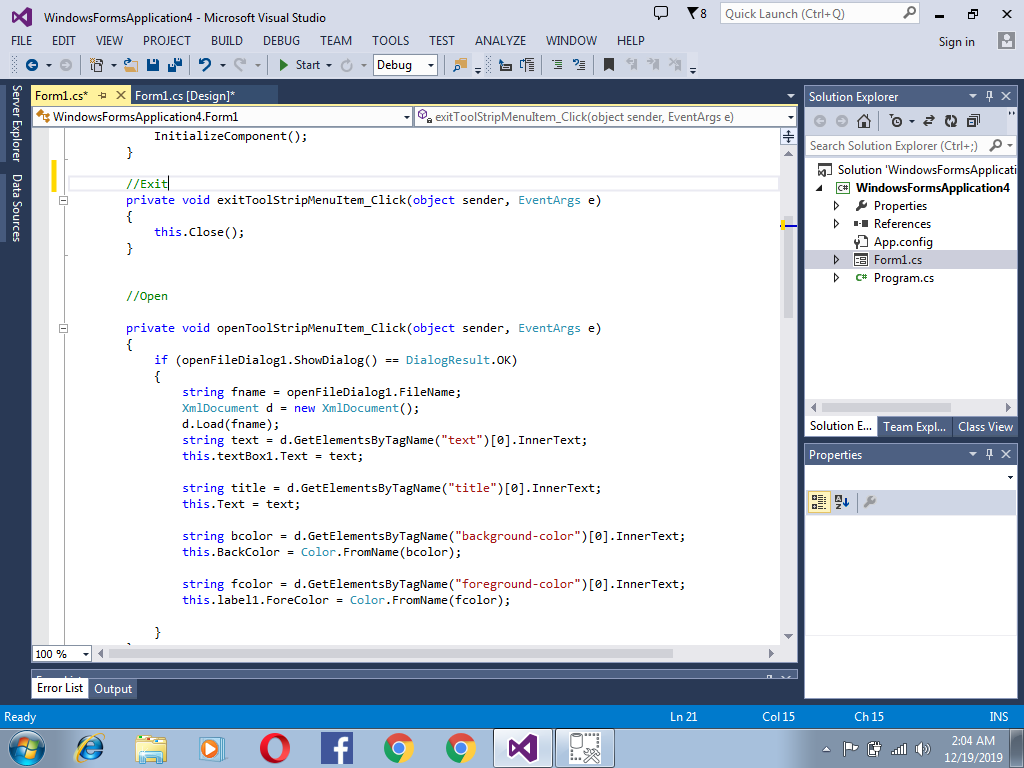
**Lab Tasks:**

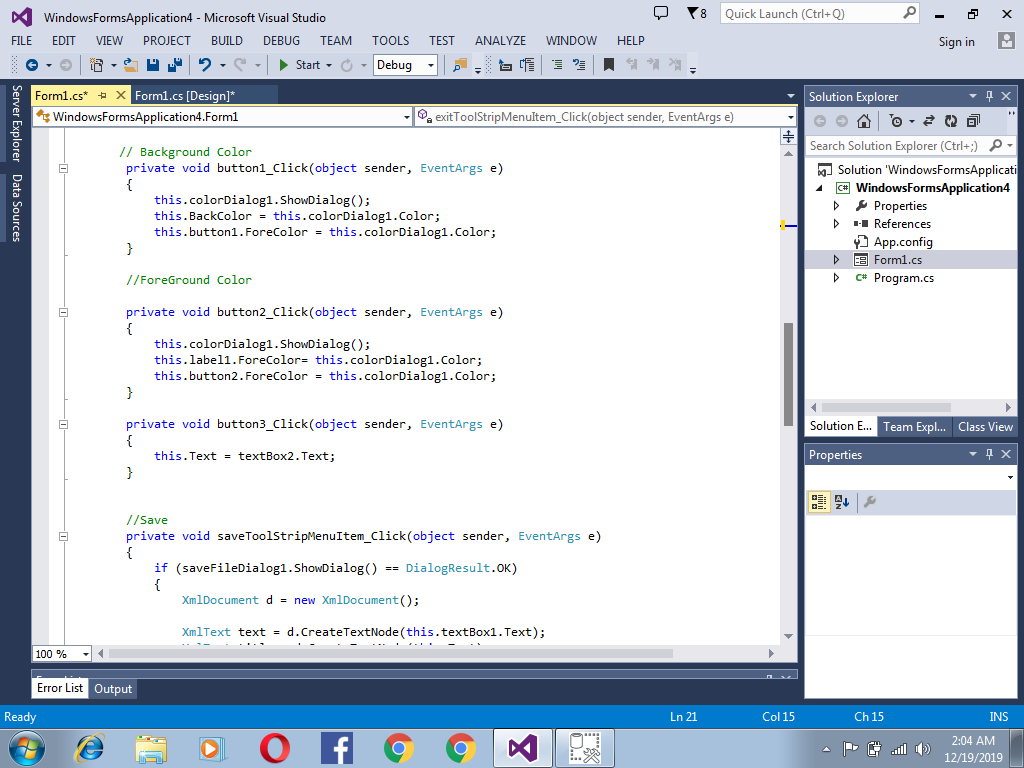
1. Create a form with a text field centered on window.
2. Create a menu strip to load XML configuration file.
3. Create an XML file to store the following information:

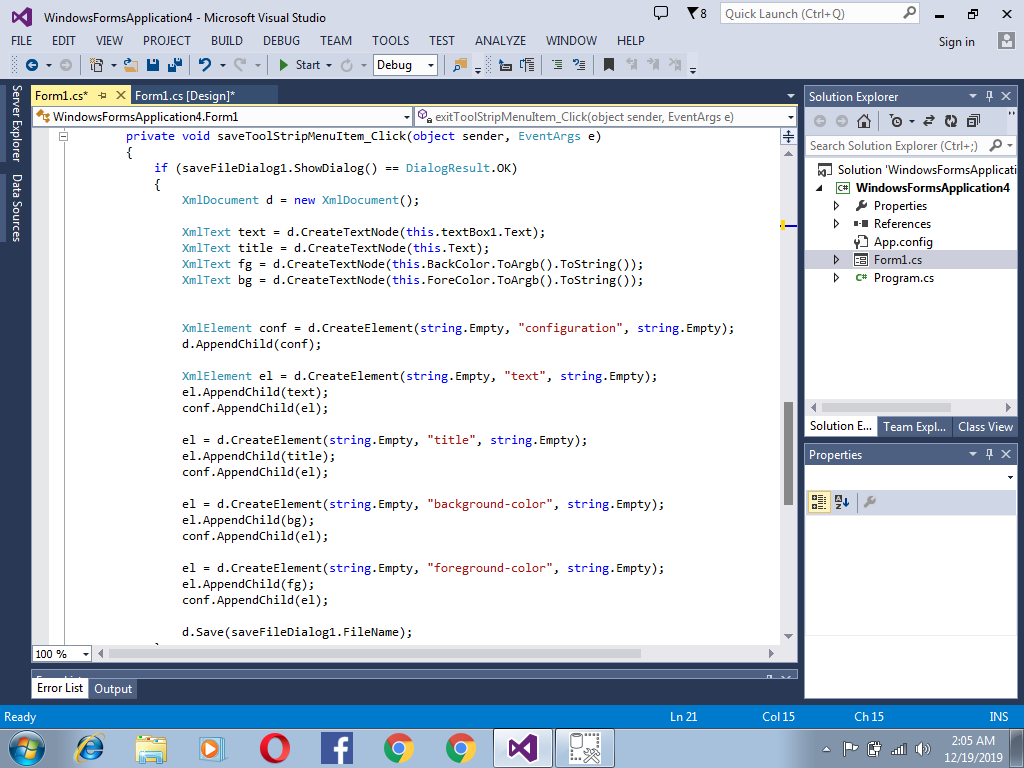
* Window size: The size of the current window
* Window title:The title of the window
* Background color: The background color of the window
* Foreground color: The foreground color of the window

1. You should be able to apply the settings stored in XML file to the window.

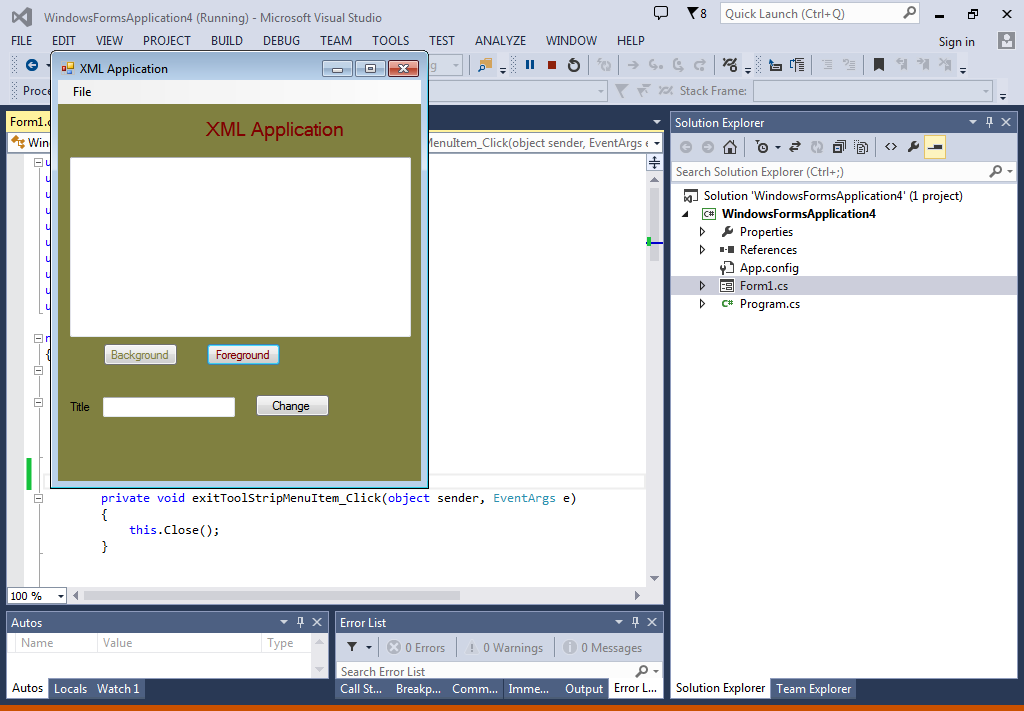








**OUTPUT:**

****

**Lab 8: To study and implement WPF and its layouts in C#**

In this lab, we will try to implement user interfaces using WPF and its layouts. We will also work with WPF data grid control.

**Lab Task:**

* 1. Design the following user interface in WPF using its various layouts.
  2. User should be able to add a new student as well as update and delete.
  3. The center data grid control should be resized when the window is maximized.

<Windowx:Class="WpfApplication1.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

Title="MainWindow" Height="350" Width="525"

Loaded ="Window\_Loaded\_1" >

<Grid>

<DockPanel>

<GridDockPanel.Dock="Top"HorizontalAlignment="Left">

<Button Content="update" Margin="355,57,47,43" ></Button>

<Button Content="cancel" Margin="355,90,47,10"></Button>

<Label Content="id"HorizontalAlignment="Left" Margin="103,25,0,0"VerticalAlignment="Top"/>

<Label Content="age"HorizontalAlignment="Left" Margin="103,77,0,0"VerticalAlignment="Top"/>

<Label Content="name"HorizontalAlignment="Left"VerticalAlignment="Top" Margin="103,51,0,0"/>

<TextBoxHorizontalAlignment="Left" Height="23" Margin="166,51,0,0"TextWrapping="Wrap" Text="TextBox"VerticalAlignment="Top" Width="120"/>

<TextBoxHorizontalAlignment="Left" Height="23" Margin="166,25,0,0"TextWrapping="Wrap" Text="TextBox"VerticalAlignment="Top" Width="120"/>

<TextBoxHorizontalAlignment="Left" Margin="166,79,0,17"TextWrapping="Wrap" Text="TextBox" Width="120"/>

</Grid>

<StackPanelHorizontalAlignment="Right"DockPanel.Dock="Bottom" Orientation="Horizontal">

<Button Margin="10,0,0,0" Content="Refresh"></Button>

<Button Margin="10,0,0,0" Content="Delete"></Button>

</StackPanel>

<DataGridx:Name="datagrid1"HorizontalAlignment="Stretch" Margin="0,0,0,0"VerticalAlignment="Stretch"SelectionChanged="DataGrid\_SelectionChanged\_1" />

</DockPanel>

</Grid>

</Window>

**Student class**

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

namespace WpfApplication1

{

classSudent

{

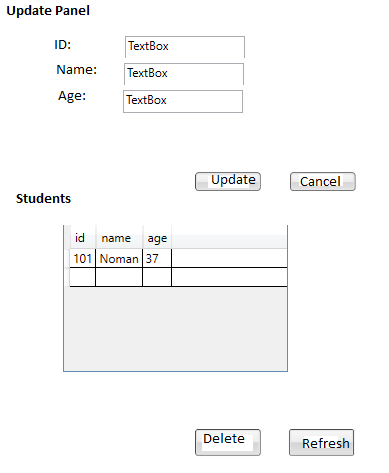
publicstringid {set; get;}

publicstringname {set; get;}

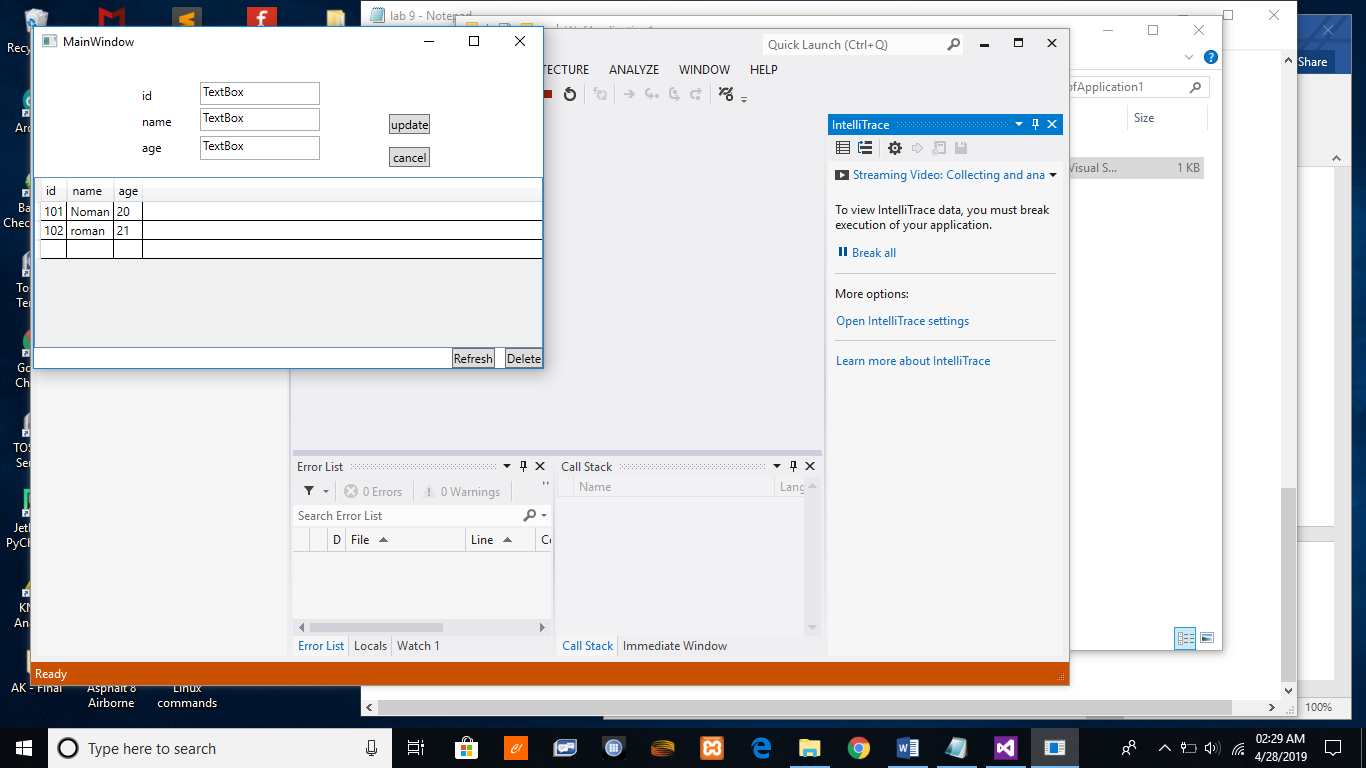
publicstringage {set; get;}

}

}



**OUTPUT:**

****

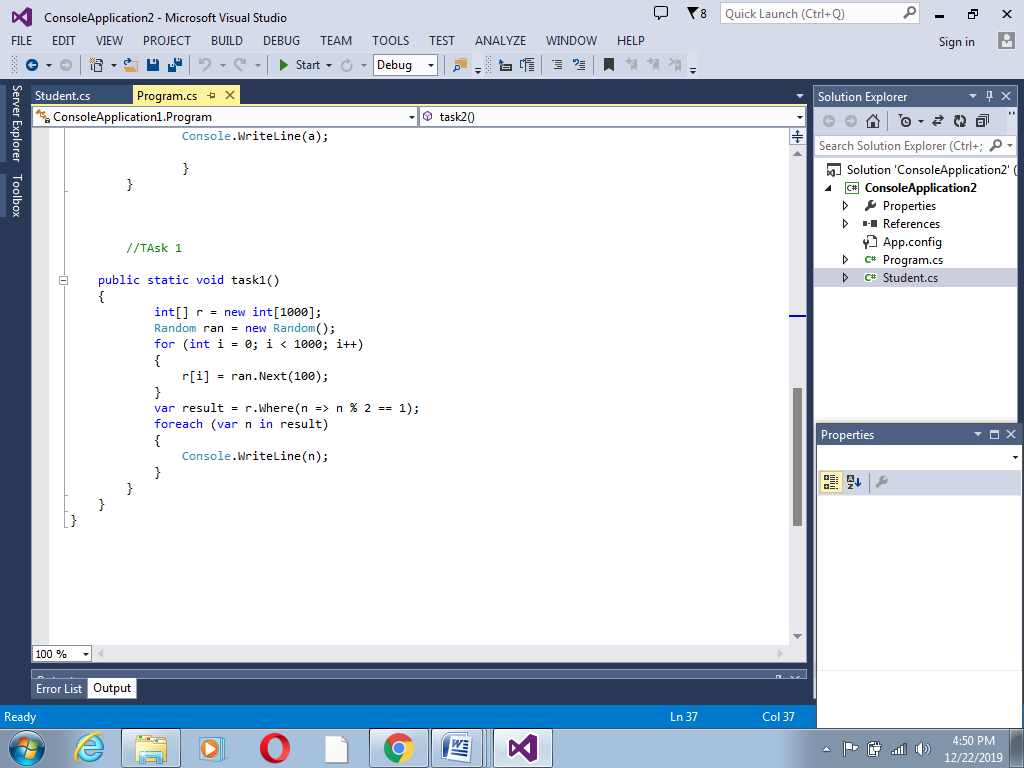
**Lab 9: To study and implement LINQ in C#**

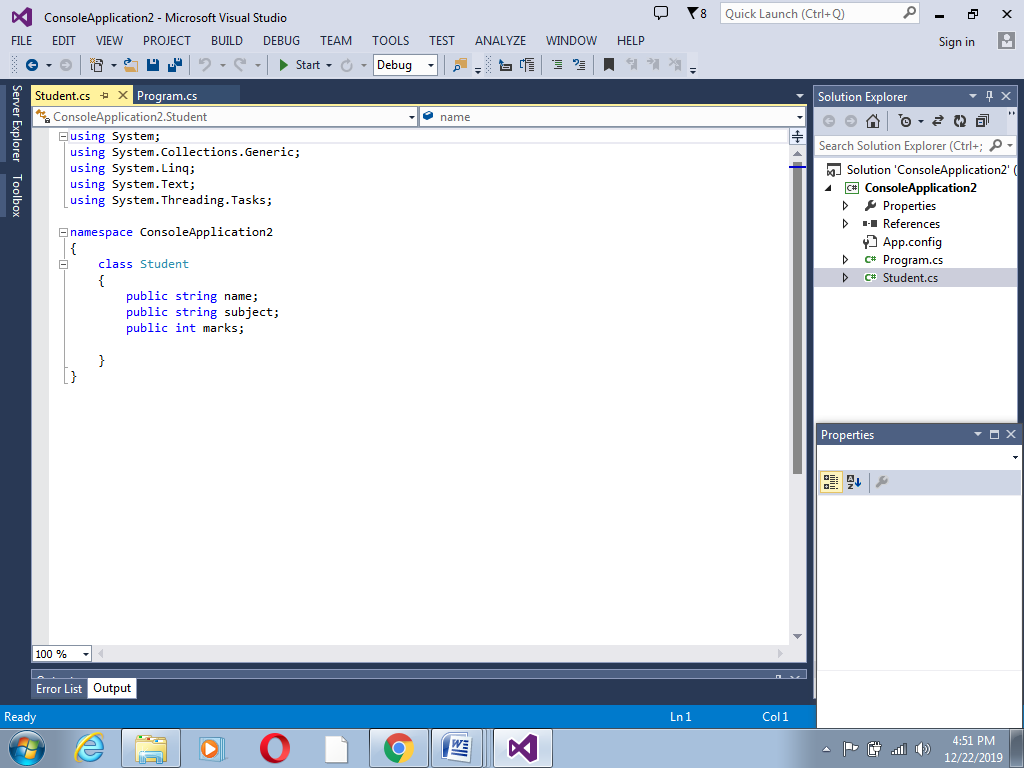
LINQ is an extension to the C# language that integrates data query directly into the programming language itself. Visual Studio 2015 and the .NET Framework 4.5 come with a number of built-in LINQ providers that provide query solutions for different types of data

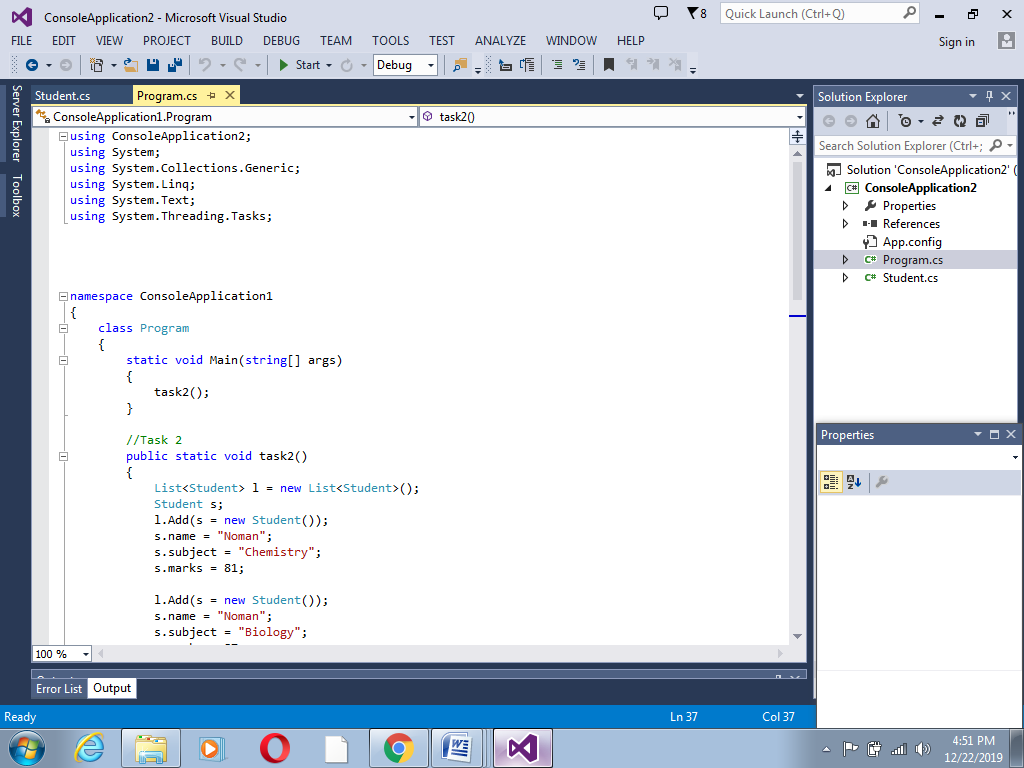
* LINQ to Objects
* LINQ to XML
* LINQ to Entities
* LINQ to Data Set

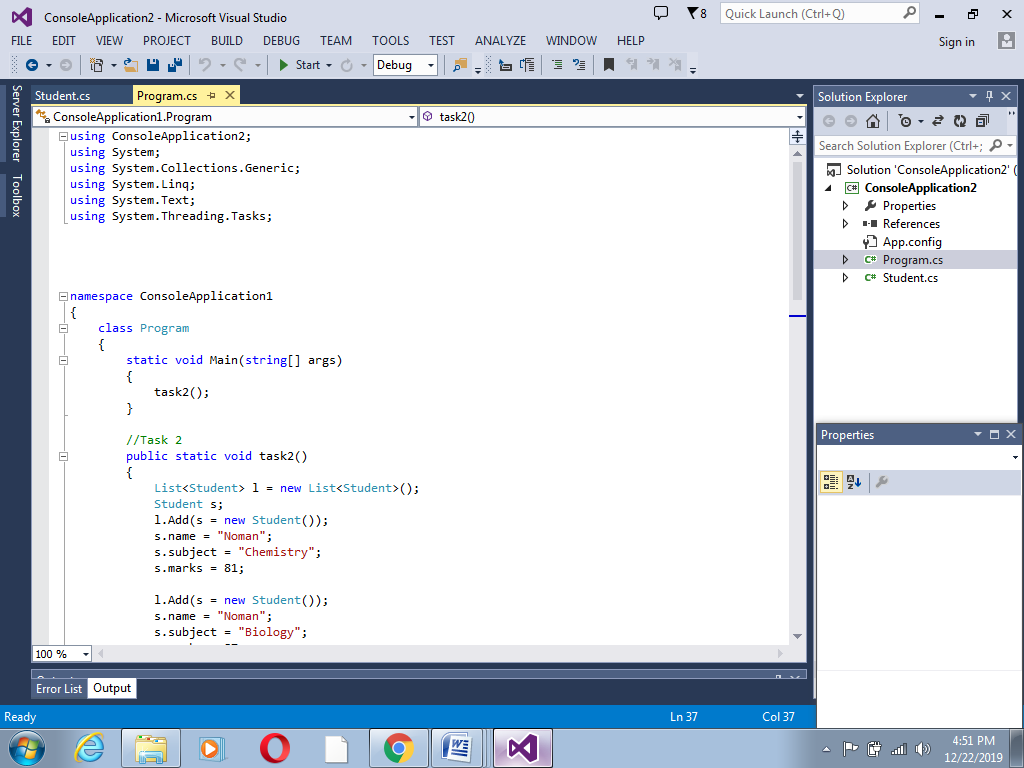
**Lab Tasks:**

1. Create an array of 1000 randomly generated numbers. Use the LINQ query to find all the odd numbers from the list.Find the count of total odd numbers.Find the maximum and minimum odd number.
2. Create a class of Student with name, subject, and marks. Now add the students in a List. Using LINQ methods and group by, find the average marks of students.
3. Create a WPF project in which you can add doctor’s details such as name, qualification and salary. The added information is saved in a List. Provide a text area through which user can write LINQ query that can be run against the list.







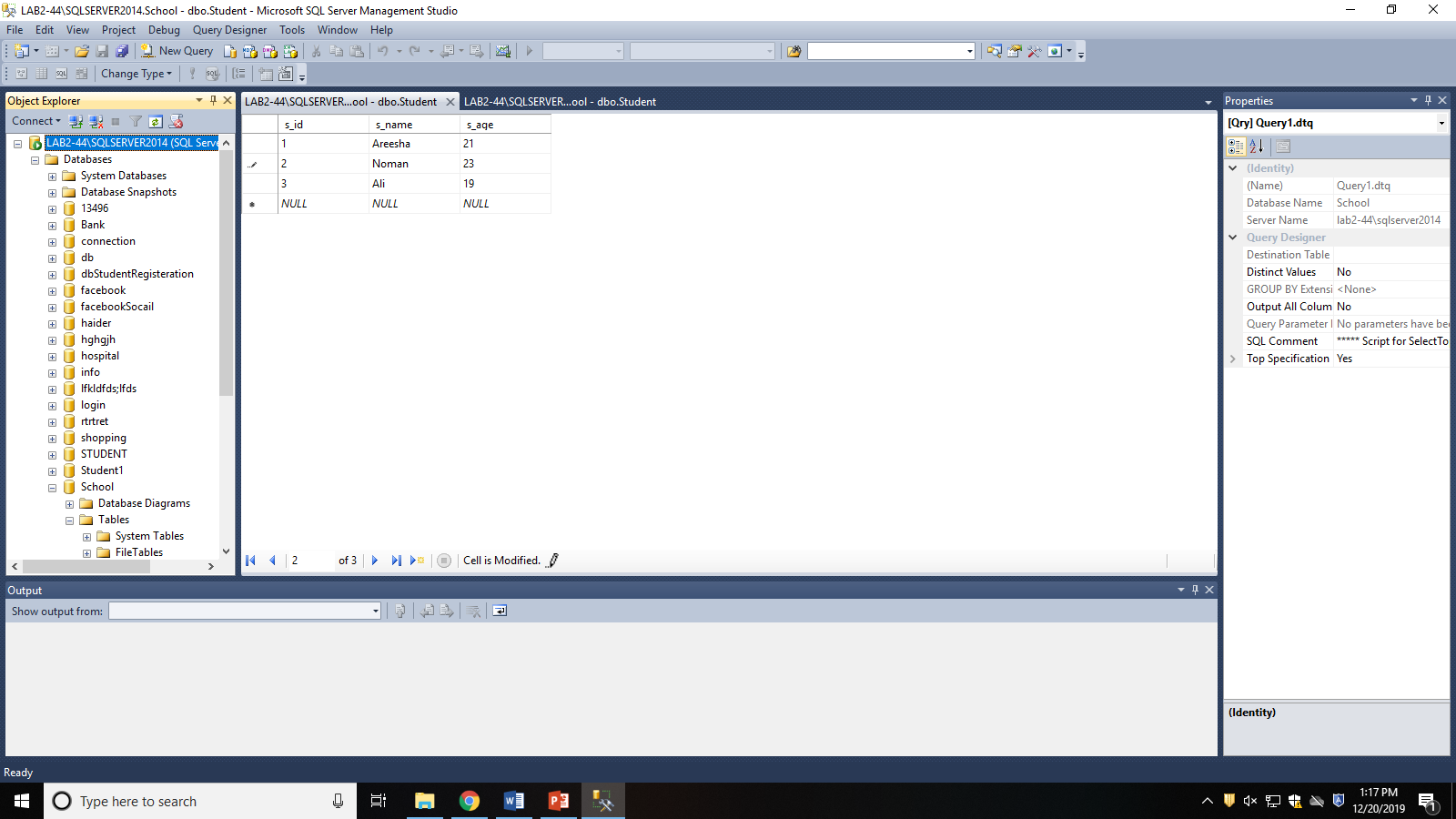


**Lab 10: To study and implement ADO .Net in C#**

The .NET platform defines a number of namespaces that allow you to interact with relational database systems. Collectively speaking, these namespaces are known as ADO.NET. In this lab, we will use ADO .net to connect to database.

**Lab Tasks:**

1. Create a database named School. Create a table Student.



1. Now create a windows form and show all the students records in a GridView
2. In the same windows form allow the user to add, update and delete a student record.

