|  |
| --- |
| **.NET PRACTICA** |
|  |
| **160470107009** |
| **Chavda foram r.** |
|  |

Contents

[AIM : Introduction to C# 1](#_Toc4759614)

[Program 1 1](#_Toc4759615)

[AIM: Inheritance 9](#_Toc4759616)

[Program 1 9](#_Toc4759617)

[Program 2 10](#_Toc4759618)

[Program 3 12](#_Toc4759619)

[Program 4 13](#_Toc4759620)

[AIM: Method & constructor overloading 16](#_Toc4759621)

[Program 1 16](#_Toc4759622)

[Program 2 21](#_Toc4759623)

[AIM: Reflection 24](#_Toc4759624)

[Program 1 24](#_Toc4759625)

[AIM: Files Operations 28](#_Toc4759626)

[Program 1 28](#_Toc4759627)

[Program 2 30](#_Toc4759628)

[Program 3 32](#_Toc4759629)

[AIM: Student Registration 36](#_Toc4759630)

[Program 1 36](#_Toc4759631)

[AIM: Validation Controls 39](#_Toc4759632)

[Program 1 39](#_Toc4759633)

[AIM: Master Page 42](#_Toc4759634)

[Program 1 42](#_Toc4759635)

Practical 1

# AIM : Introduction to C#

Variables:

Initialization

Scope

Constant

Predefined Data Types

Value Types

Reference TYpes

Flow Control

Conditional Statements(if, switch)

Loop(for, while, dowhile, foreach)

Jump(goto, break, continue, return)

Eumerations

Passing Arguments

## Program 1

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace aim

{

class Program

{

static int newint=100;

public enum TimeOfDay

{

Morning = 0,

Afternoon = 1,

Evening = 2

}

public static void Main(string[] args)

{

Console.WriteLine("\n integer types");

sbyte sb = 10;

short s = 33;

int i = 10;

long l = 33L;

byte b = 22;

ushort us = 33;

uint ul = 33u;

ulong ulo = 33ul;

Console.WriteLine("{0},{1},{2},{3},{4},{5},{6},{7}", sb, s, i, l, b, us, ul, ulo);

float f = 1.122345656767f;

double d = 12.1234455657878797;

Console.Write("\nFloat and Double:\n");

Console.WriteLine("{0} and \n{1}", f, d);

decimal dec=111.666666666666666666666M;

Console.WriteLine("decimal:\n{0} ",dec);

Console.WriteLine("\nBoolean:");

bool boolean =true;

Console.WriteLine("Status: " + boolean);

// Console.ReadLine();

char character ='d';

Console.WriteLine(character);

character = '\0';

Console.WriteLine("Now null: " + character);

object o1 = "Hi, I am ALICE";

object o2 = 15.3454365;

string strObj = o1 as string;

Console.WriteLine(strObj);

Console.WriteLine(o1.GetHashCode() + " " + o1.GetType());

Console.WriteLine(o2.GetHashCode() + " " + o2.GetType());

Console.WriteLine(o1.Equals(o2));

string s1, s2;

s1 = "this is string";

s2 = s1;

Console.WriteLine("S1 is: {0} and s2 is {1}", s1, s2);

s2 = "other string";

Console.WriteLine("S1 is: {0} and s2 is {1}", s1, s2);

s1 = "c:C:\\Users\\Dell\\source\\repos\\aim";

Console.WriteLine(s1);

s1 = @"c:C:\Users\Dell\source\repos\aim\aim";

Console.WriteLine(s1);

s1 = @"We can also write

like this";

Console.WriteLine(s1);

bool isZero;

Console.WriteLine("\nFlow Control: (if)\ni is " + i);

if (i == 10)

{

isZero = true;

Console.WriteLine("i is Zero {0}",isZero);

}

else

{

isZero = false;

Console.WriteLine("i is Non - zero");

}

int integerA = 1;

Console.WriteLine("\nSwitch:");

switch (integerA)

{

case 1:

Console.WriteLine("integerA = 1");

break;

case 2:

Console.WriteLine("integerA = 2");

//goto case 3;

break;

case 3:

Console.WriteLine("integerA = 3");

break;

default:

Console.WriteLine("integerA is not 1, 2, or 3");

break;}

WriteGreeting(TimeOfDay.Morning);

Console.WriteLine("Argument is: {0}",args[1]);

void WriteGreeting(TimeOfDay timeOfDay)

{

switch (timeOfDay)

{

case TimeOfDay.Morning:

Console.WriteLine("Good morning!");

break;

case TimeOfDay.Afternoon:

Console.WriteLine("Good afternoon!");

break;

case TimeOfDay.Evening:

Console.WriteLine("Good evening!");

break;

default:

Console.WriteLine("Hello!");

break;

}}

Console.WriteLine("Scope of Variables.\n1:");

int newint=0;

int j;

for (/\*int\*/ j = 0; j < 2; j++) //removing comment from for loop will raise error

{

//int j;

//uncomment above line to error "A local variable named 'j' cannot be declared in this

//scope because it would give a different meaning to 'j', which is already

//used in a 'parent or current' scope to denote something else"

Console.Write("{0} {1}\n", newint, Program.newint);

}

Console.WriteLine("2:");

for (int k = 0; k < 3; k++)

{

Console.Write("{0} ", k);

}//Scope of k ends here

Console.Write("\n");

//Console.Write(k);

//uncomment above line to see error "The name 'k' does not exist in the current context"

for (int k = 3; k > 0; k--)

{

Console.Write("{0} ", k);

}//scope of k ends here again

Console.WriteLine("Constants");

const int valConst = 100; // This value cannot be changed.

Console.WriteLine("{0} is constant value", valConst);

//valConst = 45;

//uncomment above line to see error "The left-hand side of an assignment must be a variable, property or indexer"

//const only allow constant variables into the expression

const int valConst2 = valConst + 9 /\* + j\*/;

//remove comments from the above line to see error "The expression being assigned to 'valConst2' must be constant"

Console.WriteLine("Another Constant: {0}", valConst2);

Console.WriteLine("\nPredefined Data Types\n\nValue Types and Reference Types");

//Value Types

int vali = 2, valj = vali;

Console.WriteLine("vali is: {0} and valj is: {1}", vali, valj);

valj = 90;

Console.WriteLine("vali is: {0} and valj is: {1}", vali, valj);

//Referece Types

Vector x, y;

x = new Vector();

x.value = 3;

y = x;

Console.WriteLine("x is: {0} and y is:{1}", x.value, y.value);

y.value = 234;

Console.WriteLine("x is: {0} and y is:{1}", x.value, y.value);

//If a variable is a reference, it is possible to indicate that it does not refer to any object by setting its value to null:

y = null;

//Console.Write("Value for y is: " + y.value);

//uncomment above line to see runtime exception "System.NullReferenceException: Object reference not set to an instance of an object."

//CTS

}

public class Vector

{

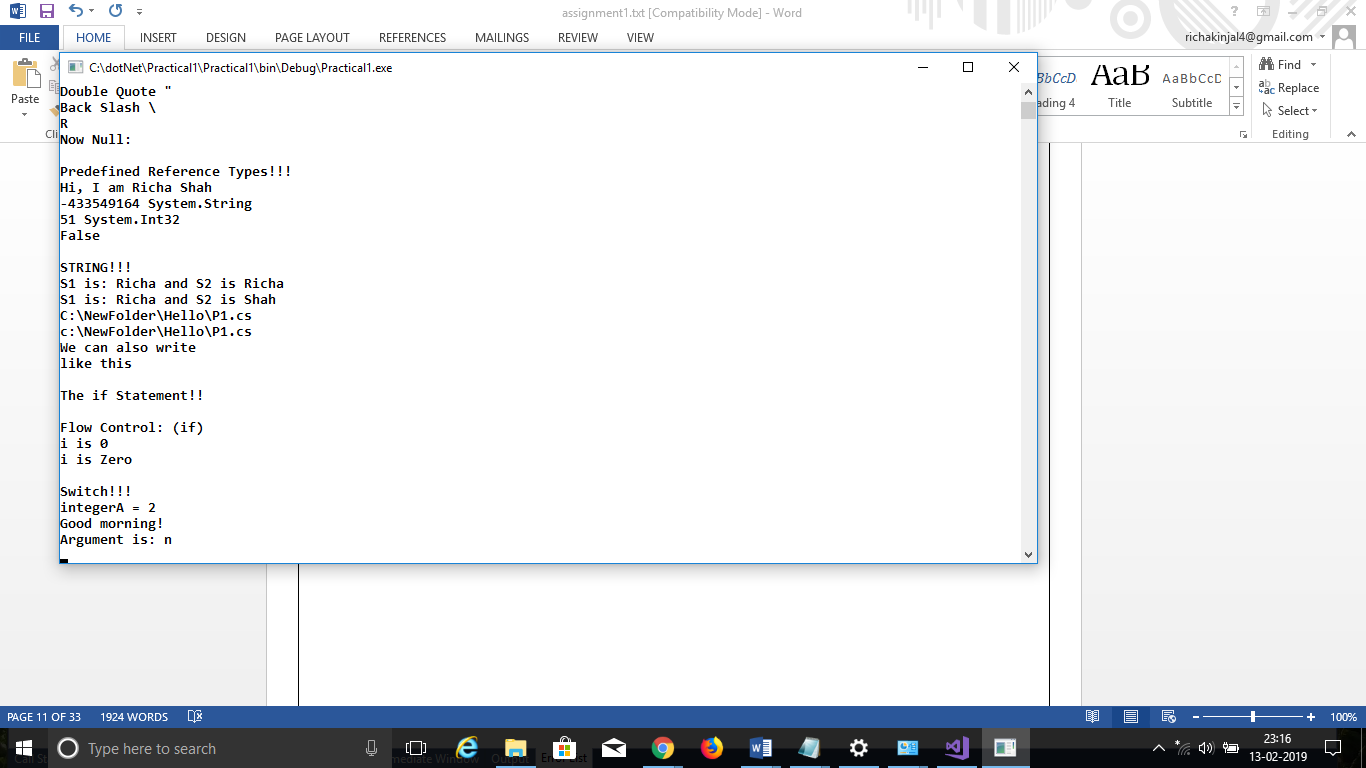
public int value;

}

}

}

***OutPut:***



Practical 2

# AIM: Inheritance

## Program 1

Perform following programs in c#.

1. Write console based program in code behind language VB or C# to print following pattern.

@ @ @ @ @

@ @ @ @

@ @ @

@ @

@

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace practical2

{

class Program

{

static void Main(string[] args)

{

for(int i=5;i>0;i--)

{

for (int j = i; j > 0; j--)

{

Console.Write("@");

}

Console.WriteLine(" ");

}

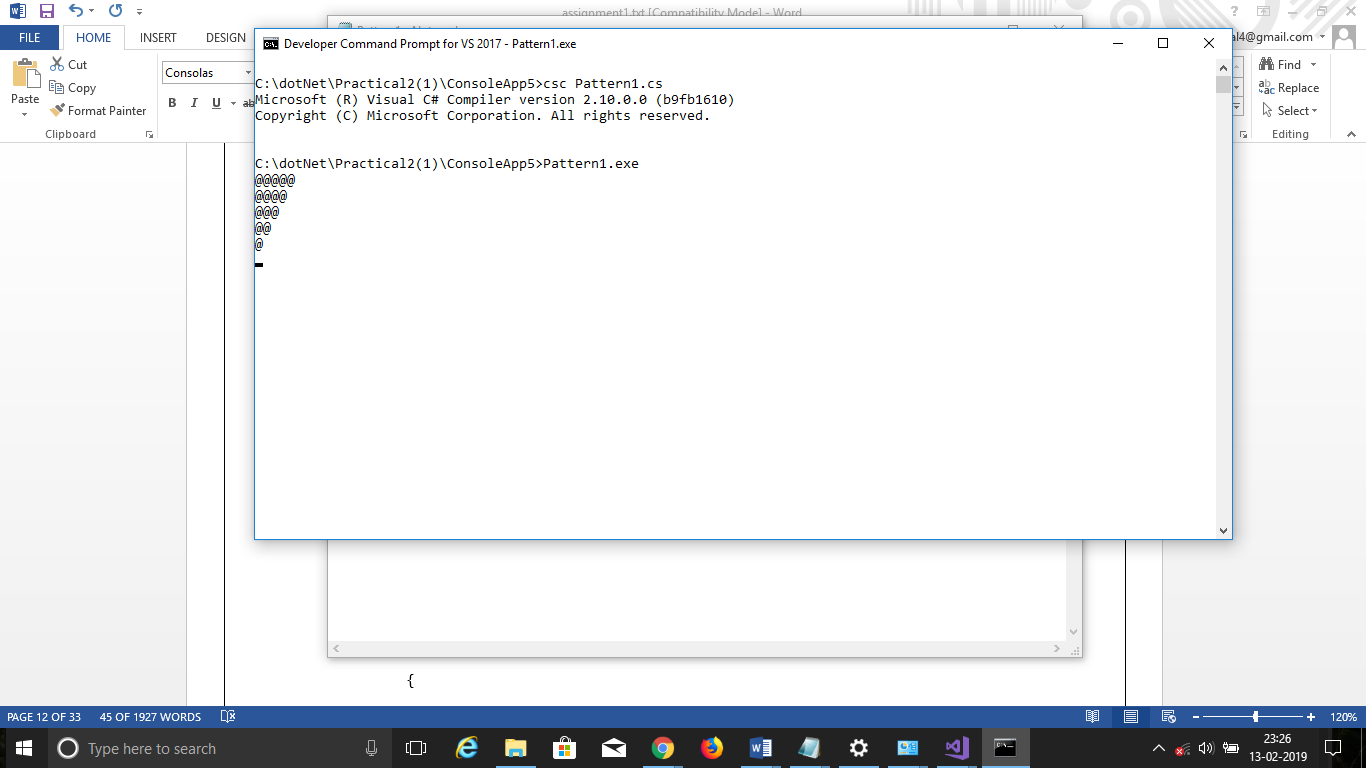
Console.ReadKey();

}

}

}

***OutPut:***



## Program 2

2. Write console based program in code behind language VB or C# to print following pattern.

1

1 2

1 2 3

1 2 3 4

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace practical2.\_1

{

class Program

{

static void Main(string[] args)

{

for(int i=1;i<5;i++)

{

for(int j=1;j<=i;j++)

{

Console.Write(j+” ”);

}

Console.WriteLine();

}

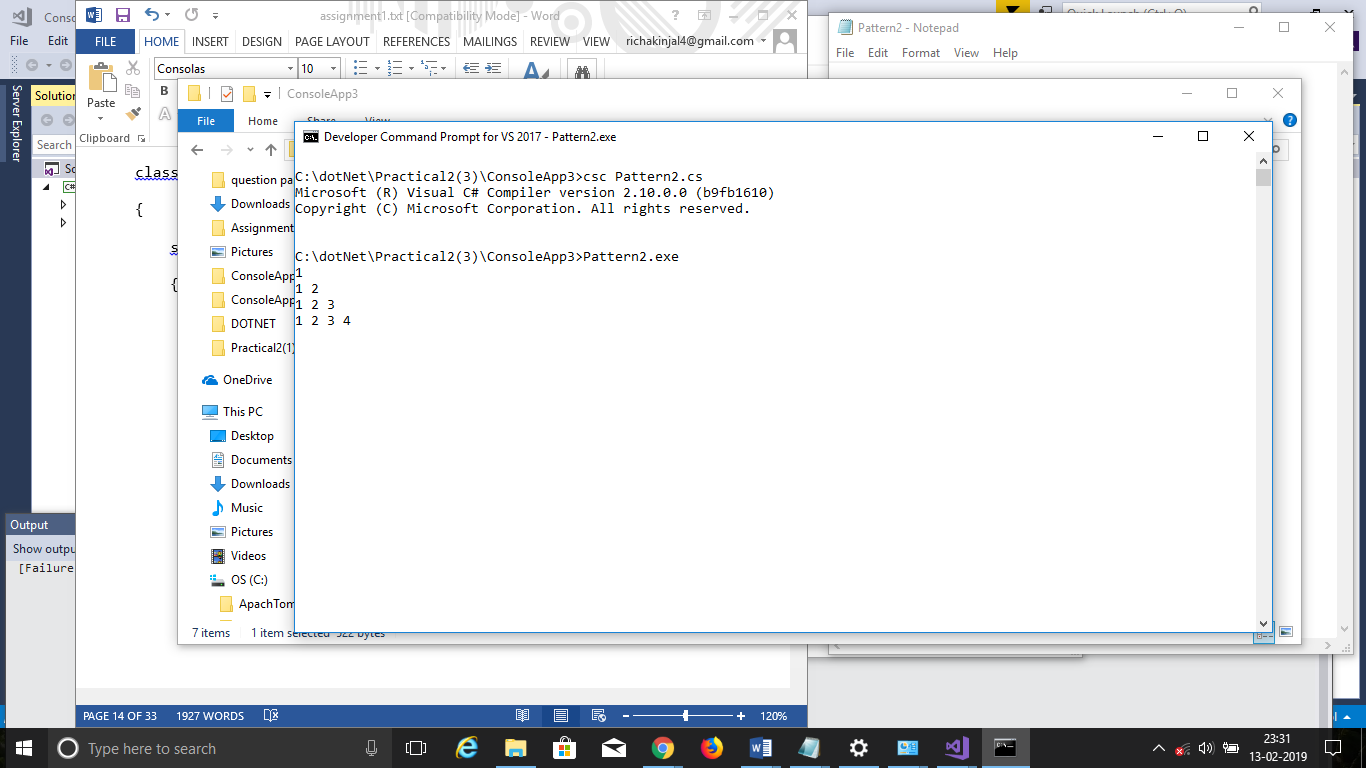
Console.ReadKey();

}

}

}

OutPut:



## Program 3

3. Write C# code to prompt a user to input his/her name and country name and then the output will be shown as an example below:

Hello Ram from country India

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace practical2.\_2

{

class Program

{

static void Main(string[] args)

{

string name;

string country;

Console.WriteLine("enter your name:");

name=Console.ReadLine();

Console.WriteLine("enter your country:");

country = Console.ReadLine();

Console.WriteLine("hello {0} from country {1}",name,country);

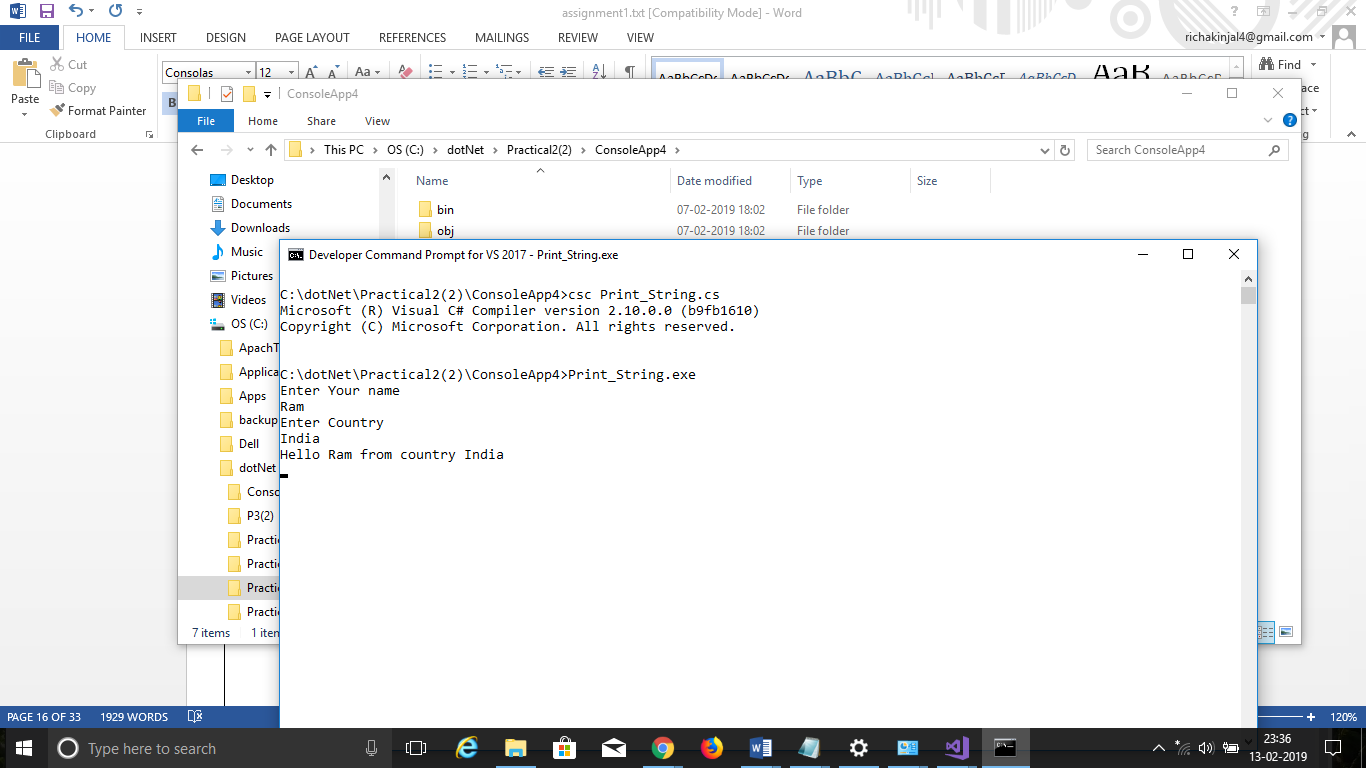
Console.ReadKey();

}

}

}

***OutPut:***



## Program 4

4. What is inheritance? Create C# console application to define Car class and derive Maruti and Mahindra from it to demonstrate inheritance.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace practical2.\_3

{

class car

{

public void Method1()

{

Console.WriteLine("this is the method of car class");

}

}

class maruti:car

{

public void method2()

{

Console.WriteLine("this is the method of maruti");

Console.ReadKey();

}

}

class mahindra:car

{

public void method3()

{

Console.WriteLine("this is the method of mahindra");

}

}

class Program

{

static void Main(string[] args)

{

mahindra m = new mahindra();

maruti m1 = new maruti();

m.Method1();

m1.Method1();

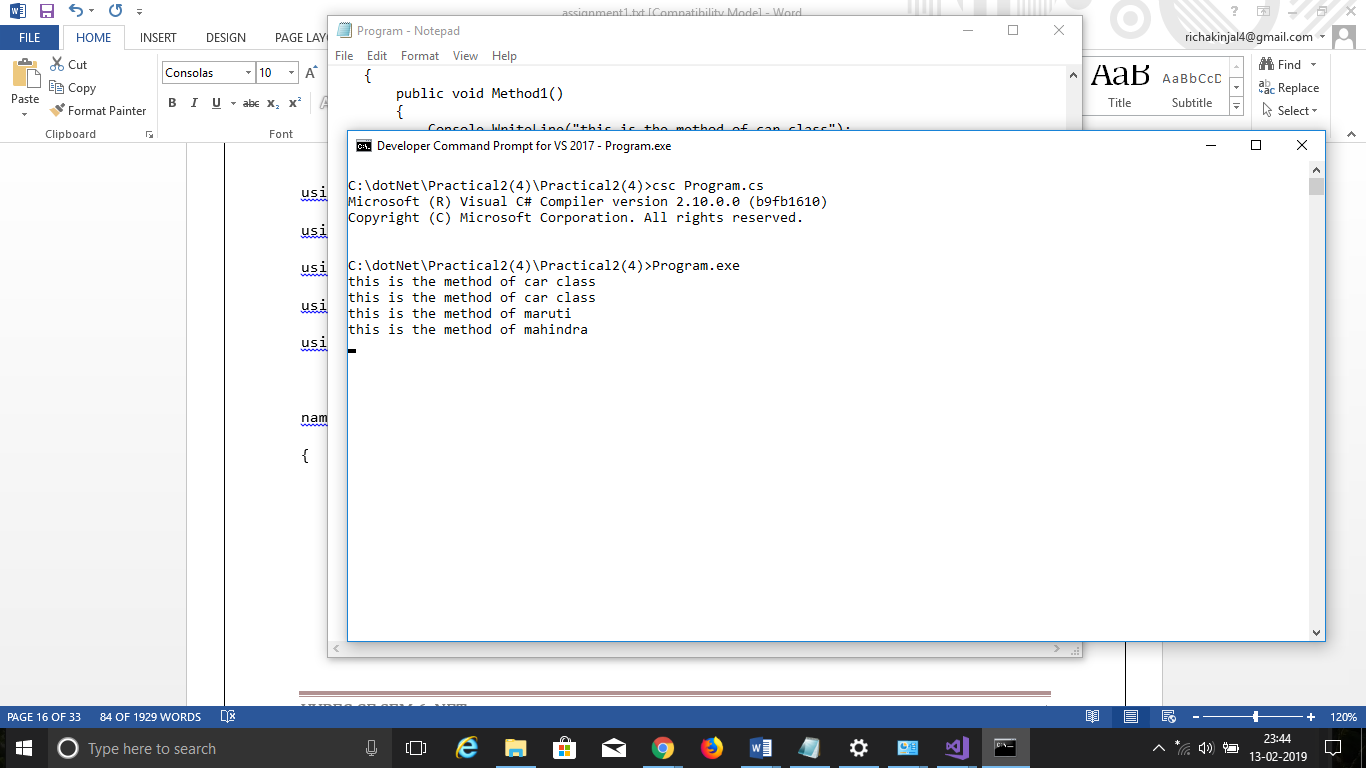
Console.ReadKey();

}

}

}

OutPut:



Practical 3

# AIM: Method & constructor overloading

## Program 1

Write a c# program to add two integers, two vectors and two metric using method overloading.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Practical3

{

class Program

{

public void add(int a, int b)

{

int sum = a + b;

Console.WriteLine("Addition is:{0}", sum);

}

public void add()

{

int i, j, n;

int[,] arr1 = new int[50, 50];

int[,] brr1 = new int[50, 50];

int[,] crr1 = new int[50, 50];

Console.Write("Input the size of the square matrix: ");

n = Convert.ToInt32(Console.ReadLine());

Console.Write("Input elements in the first matrix :\n");

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

Console.Write("{0},{1}:", i, j);

arr1[i, j] = Convert.ToInt32(Console.ReadLine());

}

}

Console.Write("Input elements in the Second matrix :\n");

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

Console.Write("{0},{1}:", i, j);

brr1[i, j] = Convert.ToInt32(Console.ReadLine());

}

}

Console.Write("\nThe First matrix is :\n");

for (i = 0; i < n; i++)

{

Console.Write("\n");

for (j = 0; j < n; j++)

Console.Write("{0}\t", arr1[i, j]);

}

Console.Write("\nThe Second matrix is :\n");

for (i = 0; i < n; i++)

{

Console.Write("\n");

for (j = 0; j < n; j++)

Console.Write("{0}\t", brr1[i, j]);

}

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

crr1[i, j] = arr1[i, j] + brr1[i, j];

}

}

Console.Write("\nAddition of Two Matrix:\n");

for (i = 0; i < n; i++)

{

Console.Write("\n");

for (j = 0; j < n; j++)

{

Console.Write("{0}\t", crr1[i, j]);

}

}

}

public void add(Vector a, Vector b)

{

Vector result=new Vector();

result.x = a.x + b.x;

result.y = a.y + b.y;

result.z = a.z + b.z;

Console.WriteLine("Addition of Two vectors is:");

Console.WriteLine("<" + result.x + "," + result.y + "," + result.z + ">");

}

static void Main(string[] args)

{

Program p = new Program();

Console.WriteLine("Value of a:");

int a = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Value of b:");

int b = Convert.ToInt32(Console.ReadLine());

p.add(a, b);

p.add();

Vector v1 = new Vector();

Vector v2 = new Vector();

// float x, y, z;

Console.WriteLine("Enter 1st vector");

Console.WriteLine("X:", v1.x);

v1.x=Convert.ToInt32( Console.ReadLine());

Console.WriteLine("Y:", v1.y);

v1.y= Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Z:", v1.z);

v1.z= Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Enter 2nd vector");

Console.WriteLine("X:", v2.x);

v2.x = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Y:", v2.y);

v2.y = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Z:", v2.z);

v2.z = Convert.ToInt32(Console.ReadLine());

p.add(v1, v2);

Console.ReadLine();

}

}

public class Vector

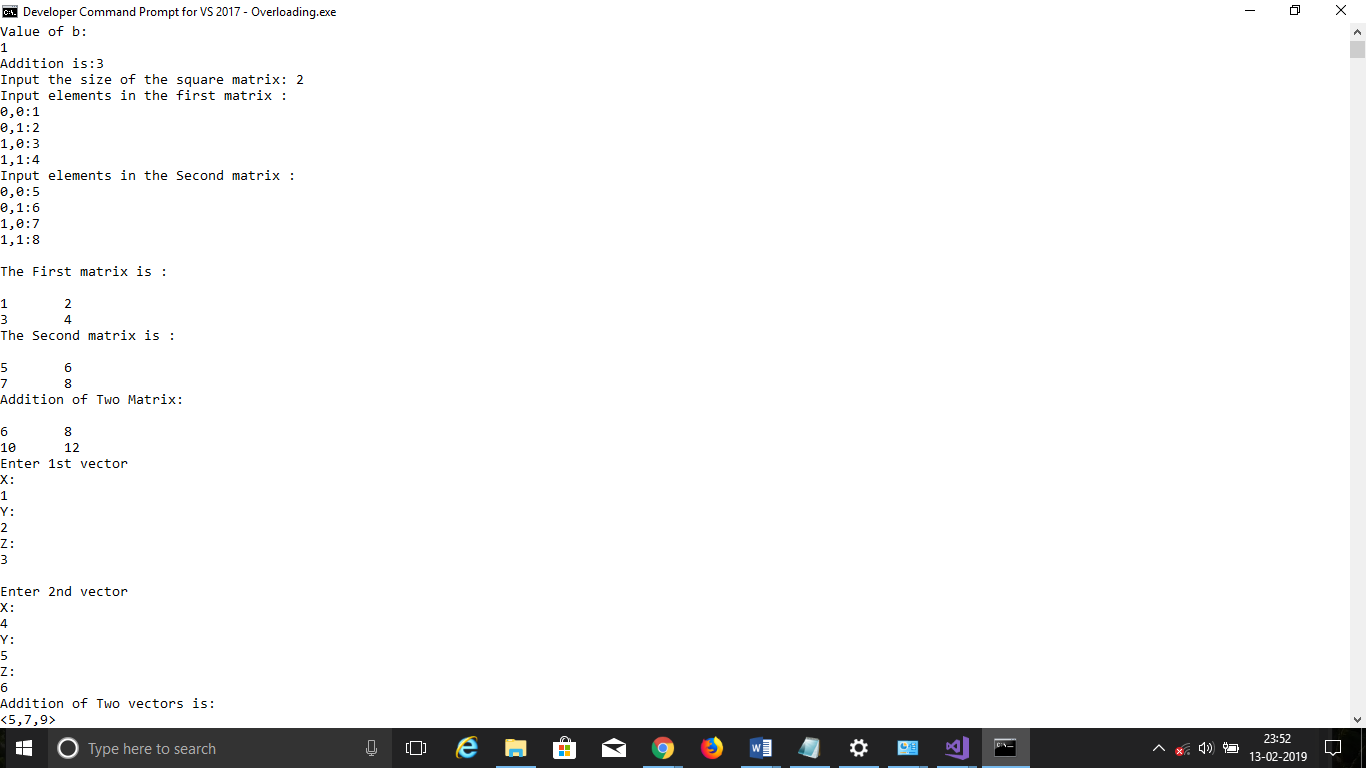
{

public float x, y,z;

}

}

***OutPut:***



## Program 2

Write a c# program that create student object. Overload constror to create new instant with following details.

1. Name

2. Name, Enrollment

3. Name, Enrollment, Branch

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace P3\_2\_

{

class Program

{

public int ID { get; set; }

public string Name { get; set; }

String name, branch;

int enroll;

Program(String Stname)

{

name = Stname;

Console.WriteLine("1st Constructor:");

Console.WriteLine("Student Name is "+Stname);

}

Program(String Stname,String Stbranch)

{

name = Stname;

branch = Stbranch;

Console.WriteLine("2nd Constructor:");

Console.WriteLine(Stname+" is in "+Stbranch+" branch");

}

Program(String Stname, String Stbranch ,int Stenroll)

{

name = Stname;

branch = Stbranch;

enroll = Stenroll;

Console.WriteLine("3rd Constructor:");

Console.WriteLine(Stname + " is in " + Stbranch+" having "+Stenroll+" Enrollment ");

}

static void Main(string[] args)

{

Program p = new Program("richa");

Program p1 = new Program("richa","Computer");

Program p2 = new Program("richa","Computer",51);

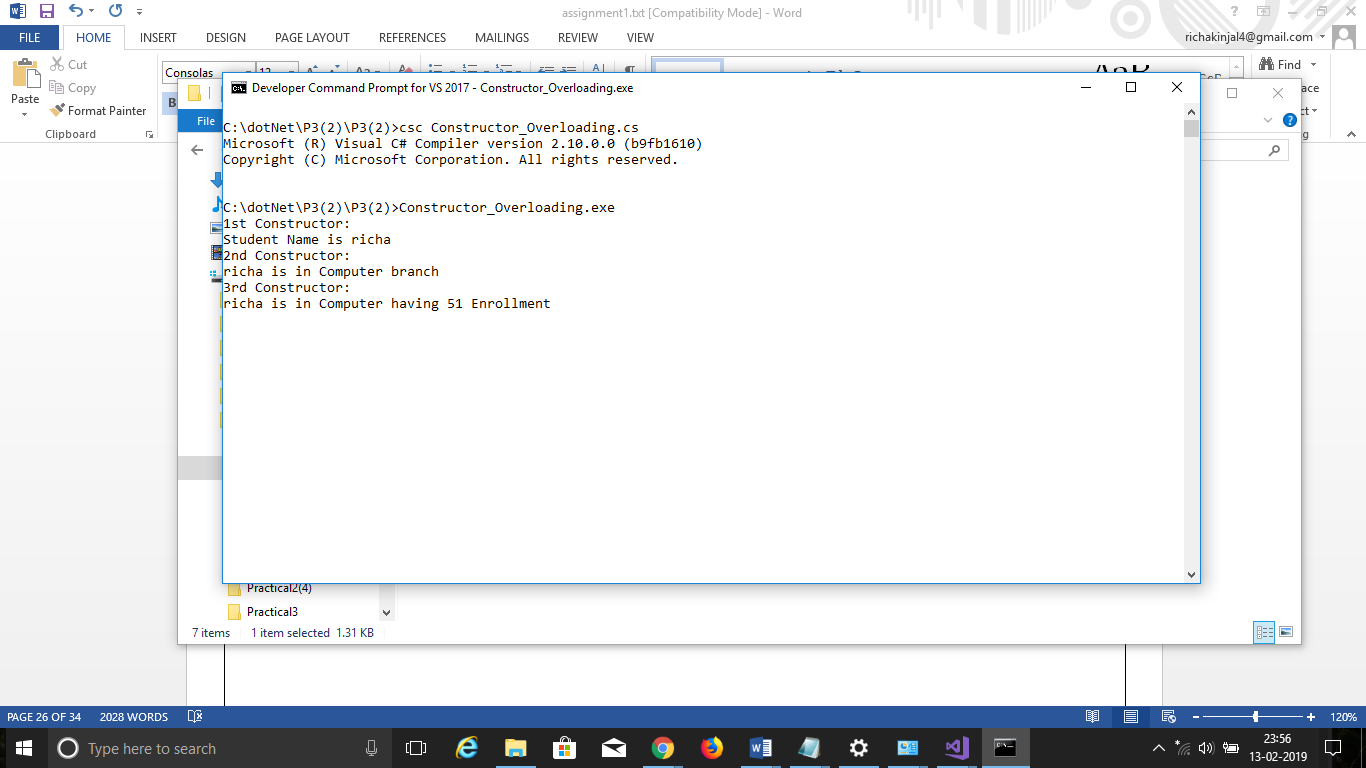
Console.ReadLine();

}

}

}

***OutPut:***



Practical 4

# AIM: Reflection

## Program 1

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Reflection;

namespace p3a1

{

class Program

{

public int ID { get; set; }

public string Name { get; set; }

String name, branch;

int enrol;

public void printID()

{

Console.WriteLine("ID is: {0}", this.ID);

}

public void printName()

{

Console.WriteLine("Name is: {0}", this.Name);

}

public Program(String name)

{

this.name = name;

Console.WriteLine("constructor 1:" + name);

}

public Program(String name, int enrol)

{

this.name = name;

this.enrol = enrol;

Console.WriteLine("constructor 2:" + name + " " + enrol);

}

public Program(String name, int enrol, String branch)

{

this.name = name;

this.enrol = enrol;

this.branch = branch;

Console.WriteLine("constructor 3:" + name + " " + enrol + " " + branch);

}

static void Main(string[] args)

{

try

{

Type T = Type.GetType("p3a1.Program");

MethodInfo[] methods = T.GetMethods();

foreach (MethodInfo method in methods)

{

Console.WriteLine(method.ReturnType + " " + method.Name);

}

PropertyInfo[] properties = T.GetProperties();

Console.WriteLine("\nProperties");

foreach (PropertyInfo property in properties)

{

Console.WriteLine(property.PropertyType + " " + property.Name);

}

Console.WriteLine("\nConstructors");

ConstructorInfo[] constructors = T.GetConstructors();

foreach (ConstructorInfo constructor in constructors)

{

Console.WriteLine(constructor.ToString());

}

Program p1 = new Program("bob");

Program p2 = new Program("bob", 1);

Program p3 = new Program("bob", 1, "computer");

Console.ReadLine();

catch (Exception e)

{

Console.WriteLine(e);

Console.ReadLine();

}

}

}

}

Practical 5

# AIM: Files Operations

## Program 1

1. Write a C# program to copy data from one file to another using StreamReader and StreamWriter class.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Practical5

{

class Program

{

static void Main(string[] args)

{

CopyFile cp = new CopyFile();

String file1= @"C:\dotNet\file1.txt";

String file2 = @"C:\dotNet\richa\file2.txt";

cp.copyFile(file1, file2);

}

}

public class CopyFile

{

public void copyFile(String file1,String file2)

{

using (StreamReader reader = new StreamReader(file1))

{

using (StreamWriter writer = new StreamWriter(file2))

{

String line = null;

while ((line = reader.ReadLine()) != null)

{

writer.WriteLine(line);

}

}

}

}

}

}

***OutPut:***

## Program 2

2. Write a C# Program to Read Lines from a File until the End of File is Reached

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Practical5\_1\_

{

class Program

{

static void Main()

{

StreamReader reader = new StreamReader("teststream.txt");

using (reader)

{

int lineNumber = 0;

String line = reader.ReadLine();

while(line!=null)

{

lineNumber++;

Console.WriteLine("Line {0}:{1}", lineNumber, line);

line = reader.ReadLine();

}

Console.ReadLine();

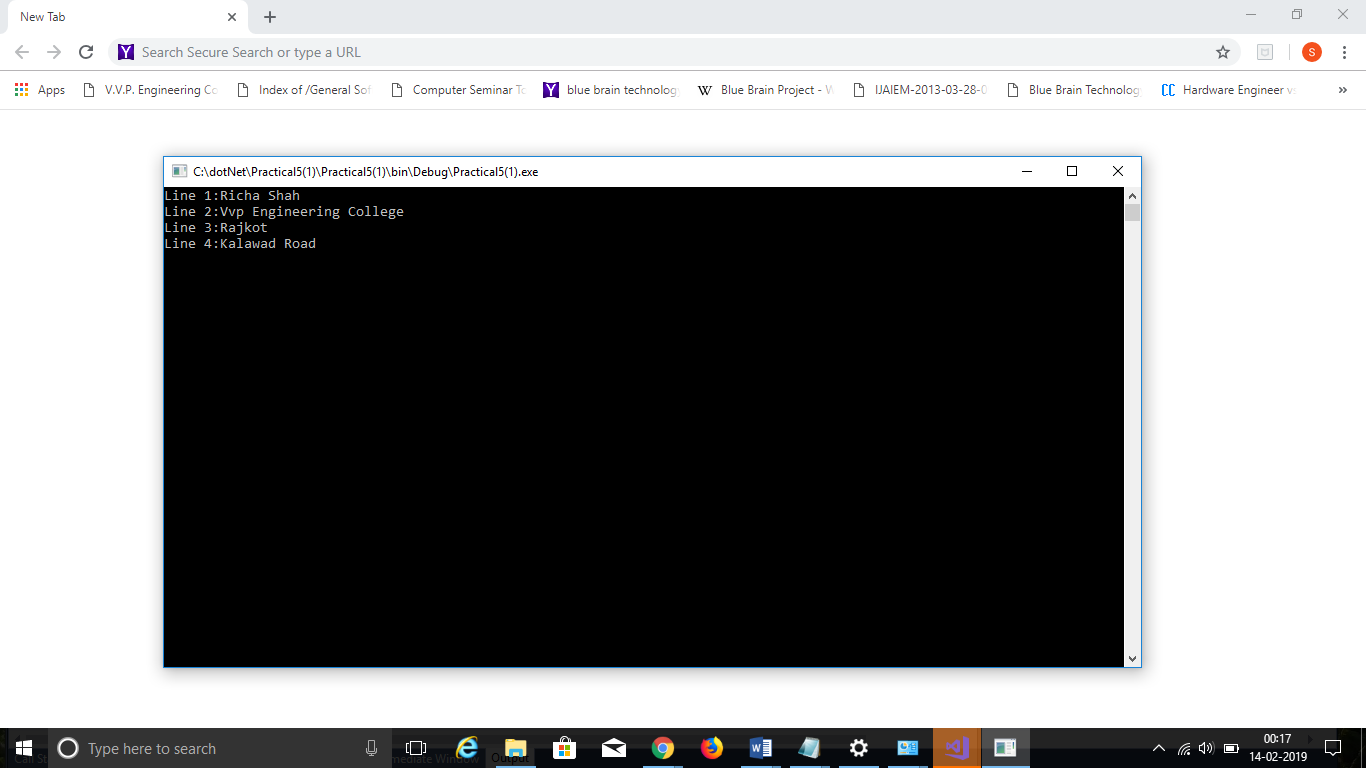
}

}

}

}

***Output:***



## Program 3

3. Write a C# Program to List Files in a Directory.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Practical5\_2\_

{

class Program

{

static void Main(string[] args)

{

string[] Directories = Directory.GetDirectories(@"C:\Users\RICHA\source\repos");

Console.WriteLine("All the Directories are:");

foreach (string dir in Directories)

{

//Console.WriteLine("All the Directories are:");

Console.WriteLine(dir);

}

string[] files = Directory.GetFiles(@"C:\Users\RICHA\source\repos");

Console.WriteLine("All the Files are:");

foreach (string file in files)

{

// Console.WriteLine("All the Files are:");

Console.WriteLine(file);

}

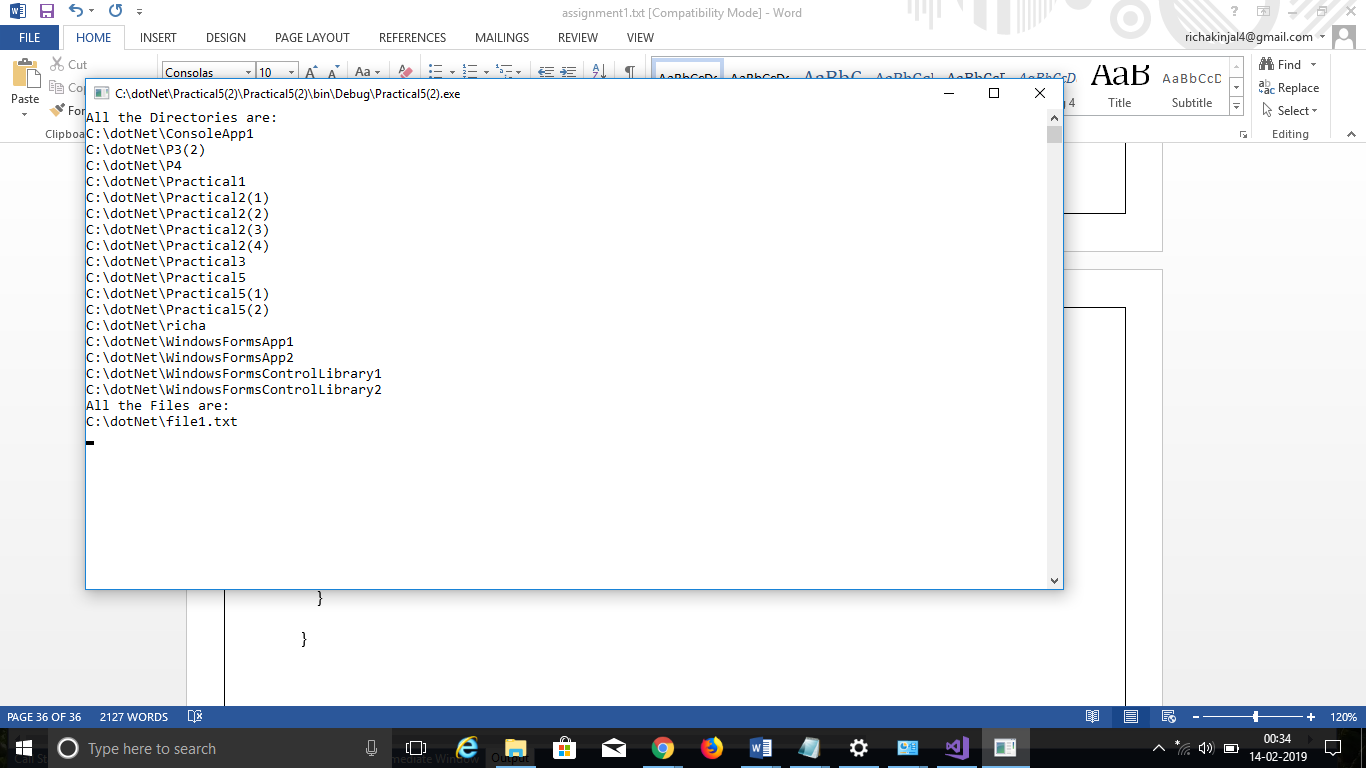
Console.ReadLine();

}

}

}

***Output:***



Practical 6

# AIM: Student Registration

## Program 1

Create Windows Form Application for Student Registration and store student Details in DataBase.

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Data.SqlClient;

using System.IO;

namespace P6\_form\_

{

publicpartialclassForm1 : Form

{

string imgPath;

public Form1()

{

InitializeComponent();

}

privatevoid label1\_Click(object sender, EventArgs e)

{

}

privatevoid Form1\_Load(object sender, EventArgs e)

{

}

privatevoid button3\_Click(object sender, EventArgs e)

{

Environment.Exit(0);

}

privatevoid button2\_Click(object sender, EventArgs e)

{

string source = @"C:\DOTNET\P6(FORM)\P6(FORM)\PROPERTIES\DATABASE1.MDF";

string select = "select count(\*) from tblStudent";

SqlConnection conn = new SqlConnection(source);

SqlCommand cmd = new SqlCommand(select, conn);

conn.Open();

int i = Convert.ToInt16(cmd.ExecuteScalar());

int textBox1 = i + 1;

string insert = "insert into tblStudent(Name,Email,Phone No,Gender,Address,imgStudent) values ( " + textBox1 + ",'" + textBox3 + "','" + textBox4 + "','" + radioButton1 + "','" + richTextBox1 + "','" + (imgPath == null ? "" : imgPath) + "' )";

cmd = new SqlCommand(insert, conn);

i = cmd.ExecuteNonQuery();

//object imgStudent = null;

if (imgPath != null)

imgStudent.Image.Save(imgPath);

MessageBox.Show("You are Done!!!");

InitializeComponent();

}

privatevoid button1\_Click(object sender, EventArgs e)

{

openFileDialog1.Filter = "Jpg|\*.jpg";

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

imgPath = @"C:\Pictures" + openFileDialog1.SafeFileName;

imgStudent.Image = Image.FromFile(openFileDialog1.FileName);

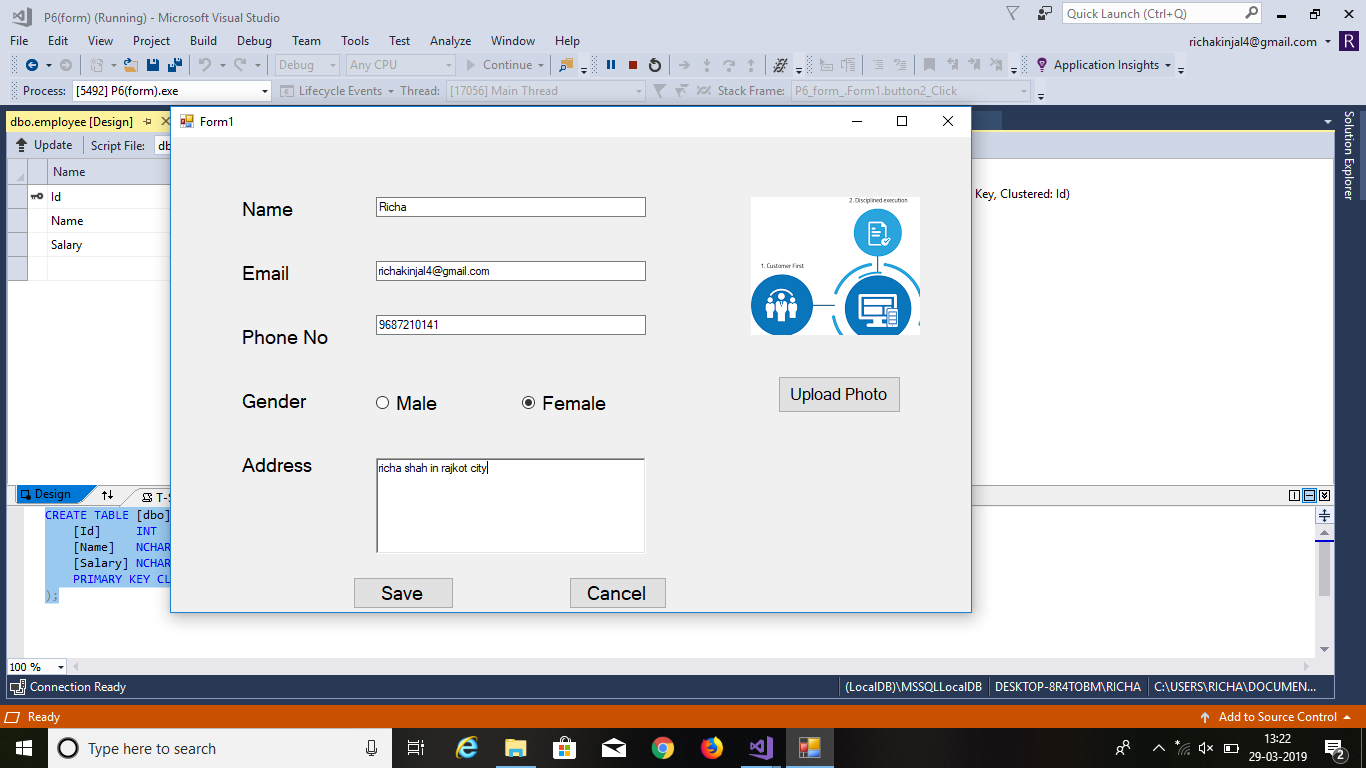
}

}

}

}

***Output:***

******

Practical 7

# AIM: Validation Controls

## Program 1

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inherits="WebApplication1.WebForm1" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<title></title>

</head>

<body style="height: 19px">

<form id="form1" runat="server">

<p>

Name:<asp:TextBox ID="txtName" runat="server" ForeColor="Red"

ToolTip="Enter Your Name"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"

ControlToValidate="txtName" Display="Dynamic" ErrorMessage="Enter Your Name"

ForeColor="Red" ToolTip="Enter Your Name">\*</asp:RequiredFieldValidator>

</p>

<p>

Email:<asp:TextBox ID="txtEmail" runat="server" ForeColor="Red"

ToolTip="Enter Your Email"></asp:TextBox>

<asp:RegularExpressionValidator ID="RegularExpressionValidator3" runat="server"

ControlToValidate="txtEmail" Display="Dynamic" ErrorMessage="Enter Valid Email"

ForeColor="Red" ToolTip="Enter Your Email"

ValidationExpression="\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*">\*</asp:RegularExpressionValidator>

</p>

<p>

Password:<asp:TextBox ID="txtPass" runat="server"></asp:TextBox>

&nbsp;&nbsp;&nbsp; Confirm Password:<asp:TextBox ID="txtConfirm" runat="server"></asp:TextBox>

<asp:CompareValidator ID="CompareValidator1" runat="server"

ControlToCompare="txtPass" ControlToValidate="txtConfirm"

ErrorMessage="Enter Same Password" ForeColor="Red"

ToolTip="Enter Same Password">\*</asp:CompareValidator>

</p>

<p>

Semester:<asp:TextBox ID="txtSem" runat="server"></asp:TextBox>

<asp:RangeValidator ID="RangeValidator1" runat="server"

ControlToValidate="txtSem" ErrorMessage="Enter Semester between 1 to 8"

ForeColor="Red" MaximumValue="8" MinimumValue="1"

ToolTip="Enter Valid Semester" Type="Integer">\*</asp:RangeValidator>

</p>

<p>

PhoneNo:<asp:TextBox ID="txtPhone" runat="server"></asp:TextBox>

<asp:RegularExpressionValidator ID="RegularExpressionValidator4" runat="server"

ControlToValidate="txtPhone" ErrorMessage="Enter Valid PhoneNo" ForeColor="Red"

ToolTip=" Enter Valid Phone Number" ValidationExpression="[0-9]{10}">\*</asp:RegularExpressionValidator>

</p>

<asp:Button ID="btnSave" runat="server" Text="Save" />

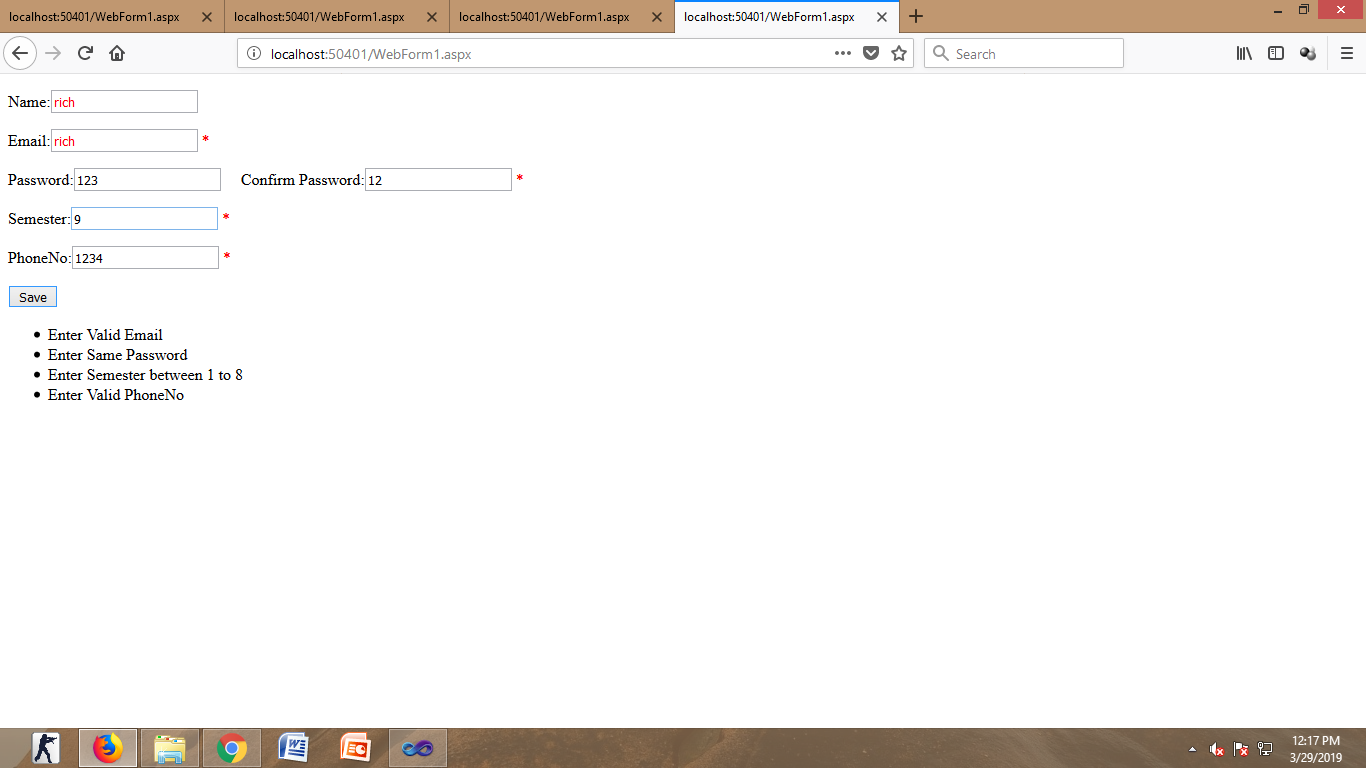
<asp:ValidationSummary ID="ValidationSummary1" runat="server" />

</form>

</body>

</html>

***Output:***

******

Practical 8

# AIM: Master Page

## Program 1

***Webform2.cs:***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

namespace WebApplication5

{

public partial class WebForm2 : System.Web.UI.Page

{

protected void Page\_Init(object sender, EventArgs e)

{

((Site1)Master).BtnSearch.Click += new EventHandler(btnSearch\_Click);

}

protected void btnSearch\_Click(object sender, EventArgs e)

{

GetData();

}

protected void Page\_Load(object sender, EventArgs e)

{

}

void GetData()

{

string source = @"Data Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\cecomp1\Documents\emp.mdf;Integrated Security=True;Connect Timeout=30;User Instance=True";

string select ="select \* from tblStudent";

SqlConnection conn = new SqlConnection(source);

SqlCommand cmd = new SqlCommand(select, conn);

conn.Open();

SqlDataReader reader = cmd.ExecuteReader();

grdEmp.DataSource = reader;

grdEmp.DataBind();

conn.Close();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

***Webform1.cs***

namespace WebApplication5

{

public partial class WebForm1 : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

}

protected void btnHeader\_Click(object sender, EventArgs e)

{

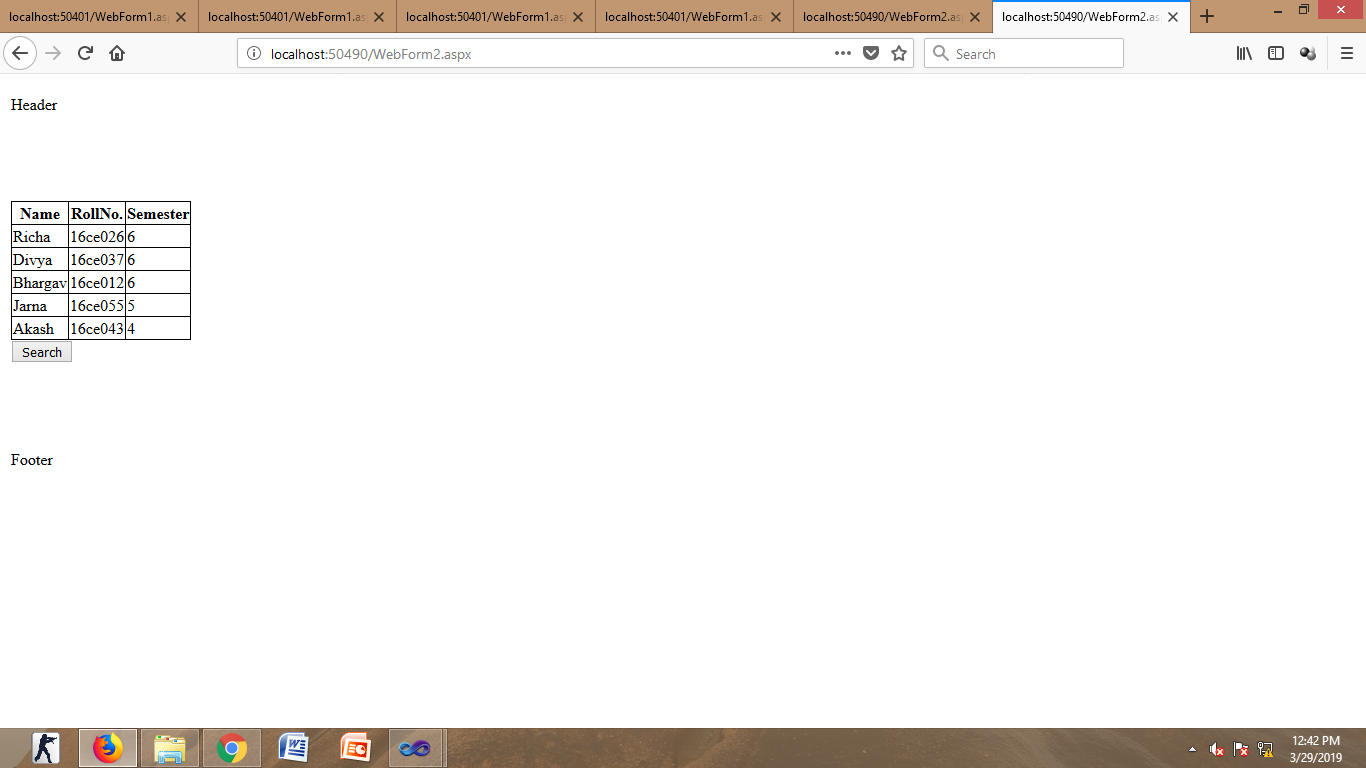
((Site1)Master).LblHeader.Text = txtHeader.Text;

}

}

}

***Output:***

******