LAB MANUAL .NET

Name:Khuman Krupali

Contents

[PRACTICAL-1 5](#_Toc4835435)

[AIM: INTRODUCTION TO C#: 5](#_Toc4835436)

[1.Introduction to c#: 5](#_Toc4835437)

[2.constant variable 5](#_Toc4835438)

[3.scope of variable 6](#_Toc4835439)

[4.scope of variable 6](#_Toc4835440)

[5.Scope of variable. 7](#_Toc4835441)

[6.consatnt variable 7](#_Toc4835442)

[7. Use of Datatypes. 8](#_Toc4835443)

[8.integer signed or unsigned variables 9](#_Toc4835444)

[9.floating variables 9](#_Toc4835445)

[10.boolean Datatype 10](#_Toc4835446)

[11.charcter Datatype 10](#_Toc4835447)

[PRACTICAL-2 11](#_Toc4835448)

[AIM: GTU PROGRAMS: 11](#_Toc4835449)

[1.Write console based program in code behind language VB or C# to print following pattern. 11](#_Toc4835450)

[@ @ @ @ @ 11](#_Toc4835451)

[@ @ @ @ 11](#_Toc4835452)

[@ @ @ 11](#_Toc4835453)

[@ @ 11](#_Toc4835454)

[@ 11](#_Toc4835455)

[2.Write console based program in code behind language VB or C# to print following pattern. 12](#_Toc4835456)

[1 12](#_Toc4835457)

[1 2 12](#_Toc4835458)

[1 2 3 12](#_Toc4835459)

[1 2 3 4 12](#_Toc4835460)

[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: 12](#_Toc4835461)

[Hello Ram from country India. 12](#_Toc4835462)

[4.Create C# console application to define Car class and derive Maruti and Mahindra from it to demonstrate inheritance. 13](#_Toc4835463)

[PRACTICAL-3 16](#_Toc4835464)

[AIM:OVERLOADING 16](#_Toc4835465)

[1.Write a c# program to add two integers, two vectors and two metric using method overloading. 16](#_Toc4835466)

[2.Write a c# program that create student object. Overload constror to create new instant with following details. 18](#_Toc4835467)

[1. Name 18](#_Toc4835468)

[2. Name, Enrollment 18](#_Toc4835469)

[3. Name, Enrollment, Branch 18](#_Toc4835470)

[PRACTICAL-4 21](#_Toc4835471)

[AIM: REFLECTION 21](#_Toc4835472)

[1.Create a c# program to find Methods, Properties and Constructors from class of running program.(Use Class from previous practical) 21](#_Toc4835473)

[PRACTICAL-5 23](#_Toc4835474)

[AIM:FILE HANDING 23](#_Toc4835475)

[1. Write a C# program to copy data from one file to another using StreamReader and StreamWriter class. 23](#_Toc4835476)

[2. Write a C# Program to Read Lines from a File until the End of File is Reached. 24](#_Toc4835477)

[3. Write a C# Program to List Files in a Directory. 25](#_Toc4835478)

[PRACTICAL-6 26](#_Toc4835479)

[AIM:WINDOWS FORM APPLICATION 26](#_Toc4835480)

[1. Create Windows Form Application for Student Registration and store student Details in Database. 26](#_Toc4835481)

[PRACTICAL-7 31](#_Toc4835482)

[AIM: ASP.NET VALIDATION CONTROL 31](#_Toc4835483)

[ RequiredFieldValidator 31](#_Toc4835484)

[ CompareValidator 31](#_Toc4835485)

[ RegularExpressionValidator 31](#_Toc4835486)

[ CustomValidator 31](#_Toc4835487)

[ RangeValidator 31](#_Toc4835488)

[ ValidationSummary 31](#_Toc4835489)

[PRACTICAL-8 33](#_Toc4835490)

[AIM:INTRODUCTION TO MASTER PAGES 33](#_Toc4835491)

# PRACTICAL-1

## AIM: INTRODUCTION TO C#:

### 1.Introduction to c#:

using System;

namespace P1

{

class MyFirstClass

{

public static void Main()

{

Console.WriteLine("HiAll");

Console.ReadKey();

return;

}

}

}

### 2.constant variable

using System;

namespace Cant

{

public class Cant

{

public static void Main()

{

int a;

a = 99;

Console.WriteLine("Value is: {0}",a);

Console.ReadKey();

}

}

}

### 3.scope of variable

using System;

namespace P1

{

class Scope1

{

public static void Main()

{

for(int i=0;i<5;i++)

{

Console.WriteLine(i);

}

for(int i=4;i>=0;i--)

{

Console.WriteLine(i);

}

}

}

}

### 4.scope of variable

using System;

namespace P1

{

class Scope2

{

public static void Main()

{

int j;

for(int i=0;i<15;i++)

{

int j;

Console.WriteLine(i);

}

}

}

}

### 5.Scope of variable.

using System;

namespace P1

{

public class Scope{

static int j = 430;

public static void Main()

{

int j =900;

Console.WriteLine(Scope.j);

}

}

### 6.consatnt variable

using System;

namespace P1

{

public class Const

{

public static void Main()

{

const double bonusPercent = 0.51;

int sal = 3000;

int bonus = (int)(sal \* bonusPercent);

Console.WriteLine(bonus);

}

}

}

### 7. Use of Datatypes.

using System;

namespace P1

{

public class Vector

{

public int value;

}

public class DataTypes

{

public static void Main()

{

int i;

int j;

i = 77;

j = i;

Console.WriteLine("i is {0} and j is {1}", i, j);

j = 20;

Console.WriteLine("i is {0} and j is {1}", i, j);

Vector x,y;

x = new Vector();

x.value = 33;

y = x;

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

y.value = 24;

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

}

}

}

### 8.integer signed or unsigned variables

using System;

namespace P1

{

class IntType

{

public static void Main()

{

sbyte sb = 33;

short s =33 ;

int i = 33;

long l = 33L;

byte b = 33;

ushort us = 33;

uint ui = 33U;

ulong ul = 33UL;

us = (ushort)ul;

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

}

}

}

### 9.floating variables

using System;

namespace P1

{

public class Floatting

{

public static void Main()

{

float f = 0.123456789F;

double d = 0.112233445566778899;

decimal dec = 11223344.1112223334445556667778889999M;

f = (float)d;

Console.WriteLine("f is {0} and d is {1} and dec is {2}", f, d, dec);

}

}

}

### 10.boolean Datatype

using System;

namespace P1

{

public class Boolean

{

public static void Main()

{

bool status = true;

Console.WriteLine(status);

}

}

}

### 11.charcter Datatype

using System;

namespace P1

{

public class Char

{

public static void Main()

{

char c = 'a';

Console.WriteLine(\a);

}

}

# PRACTICAL-2

## AIM: GTU PROGRAMS:

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

### @ @ @ @ @

### @ @ @ @

### @ @ @

### @ @

### @

using System;

namespace Pattern

{

class PatternExample

{

public static void Main()

{

int i,j=5;

for (; j > 0; j--)

{

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

Console.Write("@ ");

Console.WriteLine();

}

}

}

}

### 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;

namespace Pattern

{

class patternExample

{

public static void Main()

{

int i, j;

for (j = 1; j < 5; j++)

{

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

Console.Write(i + " ");

Console.WriteLine();

}

}

}

}

### 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;

public class userdata

{

public static void Main()

{

string name, country;

Console.Write("Enter Your Name: ");

name = Console.ReadLine();

Console.Write("Enter Your Country: ");

country = Console.ReadLine();

Console.WriteLine("Hello " + name + " from country " + country);

}

}

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

using System;

public class Car

{

protected string name;

public Car(string name)

{

this.name = name;

}

public Car()

{

}

public virtual string Name

{

get

{

return name;

}

set

{

if(value.Length>3)

name = value;

else

name="Unknown";

}

}

}

public class Maruti : Car

{

public Maruti(string name) : base(name)

{

}

public override string Name

{

get

{

return name;

}

set

{

if(value.Length>3)

name = value + " -Maruti";

else

name="Unknown";

}

}

public bool haveAGS;

}

public class Mahindra : Car

{

public Mahindra(string name) : base(name)

{

}

public Mahindra(){}

public override string Name

{

get

{

return name;

}

set

{

if(value.Length>3)

name = value + " -Mahindra";

else

name="Unknown";

}

}

}

public class Program

{

public static void Main()

{

Maruti car1 = new Maruti("Swift");

car1.haveAGS = true;

car1.Name = "Swift";

Console.WriteLine("Details Car 1: {0} and {1}",car1.Name,car1.haveAGS==true?"Have AGS":"not Have AGS");

Mahindra car2 = new Mahindra();

car2.Name = "XUV500";

Console.WriteLine("Car 2: {0}",car2.Name);

}

}

# PRACTICAL-3

## AIM:OVERLOADING

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

using System;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

namespace p3

{

public class Add

{

public void add()

{

int[,] m1 = new int[20, 20];

int[,] m2 = new int[20, 20];

int[,] m3 = new int[20, 20];

Console.WriteLine("enter size of array:");

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

Console.WriteLine("enter first array:");

for (inti = 0; i< size; i++)

{

for (int j = 0; j < size; j++)

{

m1[i, j] = Convert.ToInt32(Console.ReadLine())

}

}

Console.WriteLine("enter second array:");

for (inti = 0; i< size; i++)

{

for (int j = 0; j < size; j++)

{

m2[i, j] = Convert.ToInt32(Console.ReadLine());

}

}

for (inti = 0; i< size; i++)

{

for (int j = 0; j < size; j++)

{

m3[i, j] = m1[i, j] + m2[i, j];

}

}

Console.WriteLine("addition array:");

for (inti = 0; i< size; i++)

{

Console.Write("\n");

for (int j = 0; j < size; j++)

{

Console.Write("{0}\t", m3[i, j]);

}

Console.Write("\n");

}

}

publicint add(int a, int b)

{

return (a + b);

}

}

public class Vector

{

public void add()

{

Console.WriteLine("enter first vector");

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

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

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

Console.WriteLine("enter second vector");

int x1 = Convert.ToInt32(Console.ReadLine());

int y1 = Convert.ToInt32(Console.ReadLine());

int z1 = Convert.ToInt32(Console.ReadLine());

int x2 = x + x1;

int y2 = y + y1;

int z2 = z + z1;

Console.WriteLine("<" + x2 + "," + y2 + "," + z2 + ">");

}

}

class Program

{

static void Main(string[] args)

{

Add a1 = new Add();

Vector v1 = new Vector();

v1.add();

a1.add();

int res=a1.add(1, 2);

Console.Write("method overloading for addtion{0}",res);

Console.ReadLine();

}

}

}

### 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;

usingSystem.Collections.Generic;

usingSystem.Linq;

usingSystem.Text;

usingSystem.Threading.Tasks;

usingSystem.Reflection;

namespace p3a1

{

class Program

{

publicint ID

{

get; set;

}

public string Name

{

get; set;

}

String name, branch;

public Program(String name)

{

this.name = name;

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

}

public Program(String name, intenrol)

{

this.name = name;

this.enrol = enrol;

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

}

public Program(String name, intenrol, String branch)

{

this.name = name;

this.enrol = enrol;

this.branch = branch;

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

}

static void Main(string[] args)

{

Program p1 = new Program("bob");

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

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

Console.ReadLine();

}

}

}

# PRACTICAL-4

## AIM: REFLECTION

### 1.Create a c# program to find Methods, Properties and Constructors from class of running program.(Use Class from previous practical)

using System;

using System.Reflection;

namespace ReflectionExample

{

class MainClass

{

static void Main()

{

Type T Type.GetType("ReflectionExample.Customer");

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());

}

}

}

class Customer

{

public int ID { get; set; }

public string Name { get; set; }

public Customer(int ID, string Name)

{

this.ID = ID;

this.Name = Name;

}

public Customer()

{

this.ID = -1;

this.Name = string.Empty;

}

public void printID()

{

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

}

public void printName()

{

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

}

}

}

# PRACTICAL-5

## AIM:FILE HANDING

### 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 PRACTICAL\_5

{

class Program

{

static void Main(string[] args)

{

CopyFile cp = new CopyFile();

String file1 = @"D:\DOTNET\PRACTICAL\_5\file1.txt";

String file2 = @"D:\DOTNET\PRACTICAL\_5\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);

}

}

}

}

}

}

### 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 PRACTICAL\_5

{

class Readfile

{

static void Main()

{

StreamReader reader = new StreamReader(@"D:\DOTNET\PRACTICAL\_5\file1.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();

}

}

}

}

### 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 PRACTICAL\_5

{

class Listdir

{

static void Main(string[] args)

{

string[] Directories = Directory.GetDirectories(@"D:\DOTNET\PRACTICAL\_5");

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

foreach (string dir in Directories)

{

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

Console.WriteLine(dir);

}

string[] files = Directory.GetFiles(@"D:\DOTNET\PRACTICAL\_5");

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

foreach (string file in files)

{

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

Console.WriteLine(file);

}

Console.ReadLine();

}

}

# PRACTICAL-6

## AIM:WINDOWS FORM APPLICATION

### 1. Create Windows Form Application for Student Registration and store student Details in Database.

**Form.cs:**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

using System.Data.SqlClient;

using System.IO;

namespace StudentForm

{

public partial class Form1 : Form

{

string imgPath;

public Form1()

{

InitializeComponent();

}

private void btnsave\_Click(object sender, EventArgs e)

{

string gen = null;

string subject = null;

if (genMale.Checked == true) {

gen = "m";

}

if (genFemale.Checked == true) {

gen = "f";

}

if (ck1.Checked == true) {

subject = subject + " s1";

}

if (ck2.Checked == true) {

subject = subject + " s2";

}

string source = @"Data Source=Akash-Patel\SQLExpress;Initial Catalog=DemoDb;Integrated Security=True;Pooling=False";

string insert = "insert into tblstudent (fname,lname,gender,subject,imgStudent) values ('" + txtfname.Text + "','" + txtlname.Text + "','" + gen + "','" + subject + "','" + (imgPath

* null ? "" : imgPath) + "')"; //MessageBox.Show(insert);

//string insert = "insert into tblstudent(fname) values ('jhgjh')"; SqlConnection conn = new SqlConnection(source);

SqlCommand cmd = new SqlCommand(insert,conn); conn.Open();

int i = cmd.ExecuteNonQuery();

conn.Close();

Console.WriteLine("Success....");

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void btnimg\_Click(object sender, EventArgs e)

{

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

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

{

imgPath = openFileDialog1.SafeFileName;

pictureBox.Image = Image.FromFile(openFileDialog1.FileName); //MessageBox.Show(imgPath);

}

}

}

}

**Program.cs:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

namespace StudentForm

{

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

# PRACTICAL-7

## AIM: ASP.NET VALIDATION CONTROL

### RequiredFieldValidator

### CompareValidator

### RegularExpressionValidator

### CustomValidator

### RangeValidator

### ValidationSummary

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Validation.aspx.cs" Inherits="PRACTICAL7.Validation" %>

<!DOCTYPE html>

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

<head runat="server">

<title></title>

</head>

<body>

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

<asp:Label ID="Label1" runat="server" Text="Name"></asp:Label>

<asp:TextBox ID="txtname" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server" ControlToValidate="txtname" ErrorMessage="RequiredFieldValidator"></asp:RequiredFieldValidator>

<br />

<asp:Label ID="Label2" runat="server" Text="Password"></asp:Label>

<asp:TextBox ID="txtpwd" runat="server"></asp:TextBox>

<asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server" ControlToValidate="txtpwd" ErrorMessage="RequiredFieldValidator"></asp:RequiredFieldValidator>

<br />

<asp:Label ID="Label3" runat="server" Text="Confirm Password"></asp:Label>

<asp:TextBox ID="txtcpwd" runat="server"></asp:TextBox>

<asp:CompareValidator ID="CompareValidator1" runat="server" ControlToCompare="txtpwd" ControlToValidate="txtcpwd" ErrorMessage="CompareValidator"></asp:CompareValidator>

<br />

<asp:Label ID="Label4" runat="server" Text="Email"></asp:Label>

<asp:TextBox ID="txtemail" runat="server"></asp:TextBox>

<%--<asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server" ControlToValidate="txtemail" ErrorMessage="RegularExpressionValidator" ValidationExpression=="\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*"></asp:RegularExpressionValidator>--%>

<br />

<asp:Label ID="Label5" runat="server" Text="Age"></asp:Label>

<asp:TextBox ID="txtage" runat="server"></asp:TextBox>

<asp:RangeValidator ID="RangeValidator1" runat="server" ControlToValidate="txtage" ErrorMessage="RangeValidator" MaximumValue="30" MinimumValue="15"></asp:RangeValidator>

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

<br />

</form>

</body>

</html>

# PRACTICAL-8

## AIM:INTRODUCTION TO MASTER PAGES

**admin.master**

<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="admin.master.cs" Inherits="masternew.admin" %>

<!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>

<asp:ContentPlaceHolder ID="head" runat="server">

</asp:ContentPlaceHolder>

</head>

<body>

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

<div>

<table>

<tr>

<td colspan="2">

Header<asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>

&nbsp;</td>

</tr>

<tr>

<td>

menu

</td>

<td>

<asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">

<asp:TextBox ID="txtname" runat="server"></asp:TextBox>

<asp:Button ID="btnsave" runat="server" onclick="Btnsave\_Click" Text="Button" />

</asp:ContentPlaceHolder>

</td>

<td>

<asp:ContentPlaceHolder ID="ContentPlaceHolder2" runat="server">

</asp:ContentPlaceHolder>

</td>

</tr>

<tr>

<td>

footer

</td>

</tr>

</table>

</div>

</form>

</body>

</html>

**admin.Master.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace masternew

{

public partial class admin : System.Web.UI.MasterPage

{

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

{

}

public Button Btnsave

{

get { return btnsave; }

}

public TextBox Txtname

{

get { return txtname; }

}

}

}

**WebForm1.aspx**

<%@ Page Title="" Language="C#" MasterPageFile="~/admin.Master" AutoEventWireup="true"

CodeBehind="WebForm1.aspx.cs" Inherits="masternew.WebForm1" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">

enter name:

<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>

<asp:Button ID="Button1" runat="server" Text="Button" />

</asp:Content>

<asp:Content ID="Content3" runat="server" ContentPlaceHolderID="ContentPlaceHolder2">

enter name:

<asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>

<asp:Button ID="Button2" runat="server" Text="Button" />

</asp:Content>

WebForm1.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace masternew

{

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

{

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

{

}

}

}

**WebForm2.aspx**

<%@ Page Title="" Language="C#" MasterPageFile="~/admin.Master" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs" Inherits="masternew.WebForm2" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">

<asp:TextBox ID="txtname" runat="server"></asp:TextBox>

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

</asp:Content>

<asp:Content ID="Content3" ContentPlaceHolderID="ContentPlaceHolder2" runat="server">

<asp:GridView ID="GridView2" runat="server">

</asp:GridView>

</asp:Content>

**WebForm2.aspx.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 masternew

{

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

{

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

{

((admin)Master).Btnsave.Click += new EventHandler(Btnsave\_Click);

}

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

{

}

void GetData()

{

string source =@"Data Source=mycomputer\sqlexpress;Initial Catalog=DBstudent;Integrated Security=True;Pooling=False";

string select="select \*from tblStudent where fname like''%"+((admin)Master).Txtname.Text+"%";

SqlConnection con = new SqlConnection(source);

SqlCommand cmd = new SqlCommand(select, con);

con.Open();

SqlDataReader reader = cmd.ExecuteReader();

GridView2.DataSource = reader;

GridView2.DataBind();

con.Close();

}

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

{

GetData();

}

}