LAB MANUAL .NET

NAME:Khunt Rupali

Er.No:170473107013

Branch:Computer

Sem:6th

Table of Contents

PRACTICAL-1

AIM: INTRODUCTION TO C#:

```
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



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

            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:

```
1:

0 90

1 90

2:

0 1 2

3 2 1 Constants

100 is constant value

Another Constant: 109


Predefined Data Types


Value Types and Reference Types

vali is: 2 and valj is: 2

vali is: 2 and valj is: 90

x is: 3 and y is:3

x is: 234 and y is:234



Integer Types

33 33 33 33 33 33 33 33


Float and Double:
```

```
11.22334 and

11.2233445566779
Decimal:

111.222333444555666777888999


Boolean:
Status: True


Character:
Single Quote '

Double Quote "
Back Slash \

A

Now null:


Hi, I am an Object
-1735802816 System.String

34 System.Int
32 False
S1 is: String 1 and s2 is String 1

S1 is: String 1 and s2 is New String 1
```

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);
      }
}
```

Output:

```
Enter your name:vvp

Enter your country:india

Hello vvp from country india
```

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);
    }
}
```

Output:

```
This is maruti class


This is Mahindra class...
```

PRACTICAL-3

AIM:OVERLOADING

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();

        }

    }

}
```

```
Enter Number 1:

1

Enter Number 2:

2

Addition of Number:3


Enter Vector 1:

1

2
```

```
Enter Vector 2:

3

1
Addition of vector: x=4, y=3


Addition of two metrics:

Addition: 6
Addition: 8

Addition: 10

Addition: 12
```

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();
        }
    }
}
```

```
First Constructor initiated..

Second Constructor initiated..


Third Constructor initiated..
```

PRACTICAL-4

AIM: REFLECTION

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);
        }
    }
}
```

Output:

```
System.Int32 get_ID

System.Void set_ID

System.String get_Name

System.Void set_Name

System.Void printID
System.Void printName

System.String ToString

System.Boolean Equals

System.Int32 GetHashCode

System.Type GetType


Properties

System.Int32 ID

System.String Name


Constructors

Void .ctor(Int32, System.String)

Void .ctor()
```

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);
                    }
                }
            }
        }
    }
}
```

Output:

```
F1.txt: Hello World…

F2.txt: Hello World…
```

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();
            }
        }
    }
}
```

F1.txt:

```
Hello World.....
hii

how
are you
???
```

F2.txt:

```
Hello World.....
hii

how
are you
???
```

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();
        }
    }
```

Output:

```
E:\SEM-6 .NET\VS\P1-master

E:\SEM-6 .NET\VS\p2


E:\SEM-6 .NET\VS\Assignment.docx

E:\SEM-6 .NET\VS\C# word.txt


E:\SEM-6 .NET\VS\Doc1.docx

E:\SEM-6 .NET\VS\P1-master.zip


E:\SEM-6 .NET\VS\p1.cs


E:\SEM-6 .NET\VS\p1.exe

E:\SEM-6 .NET\VS\VS.docx


E:\SEM-6 .NET\VS\~$VS.docx
```

PRACTICAL-6

AIM:WINDOWS FORM APPLICATION

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

    {
```

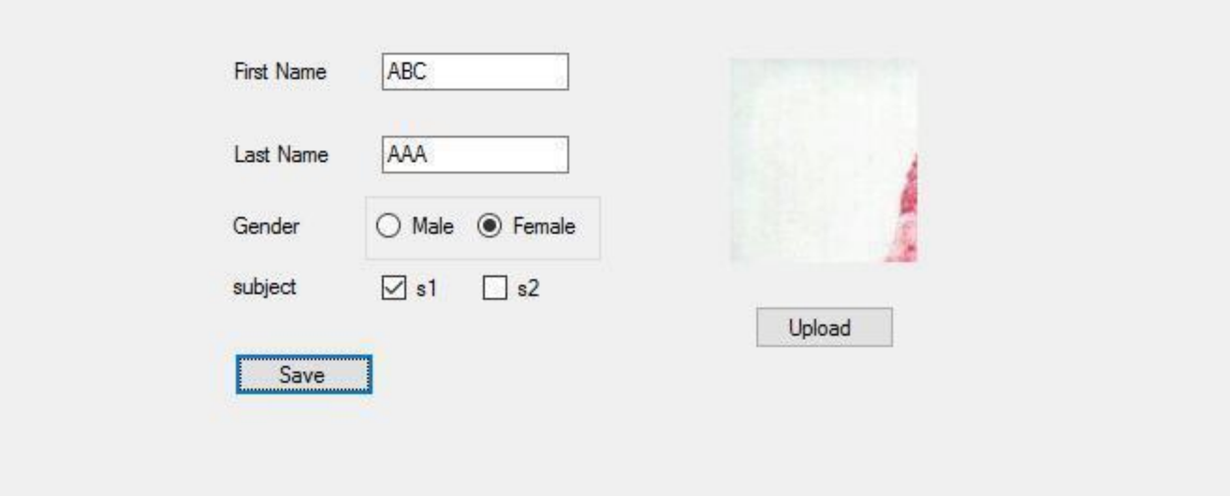
```
string imgPath;

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());
        }
    }
}
```



First Name ABC

Last Name AAA

Gender ○ Male ● Female

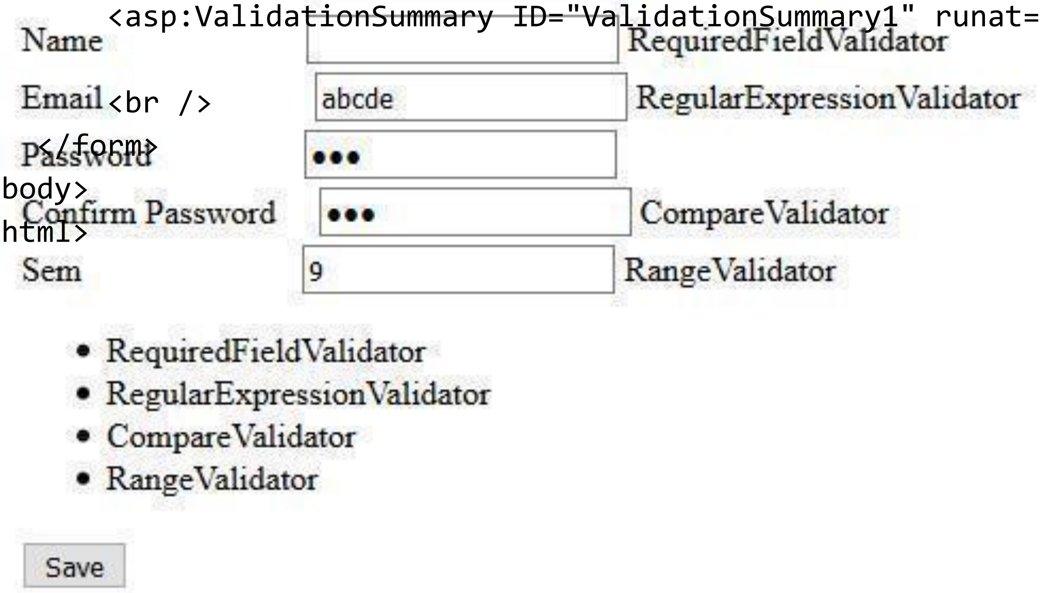
subject ☑ s1 □ s2

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>
```



Name RequiredFieldValidator

Email abcde RegularExpressionValidator

Password ●●●

Confirm Password ●●● CompareValidator

Sem 9 RangeValidator

- RequiredFieldValidator
- RegularExpressionValidator
- CompareValidator
- RangeValidator

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)
        {

        }

    }
}
```

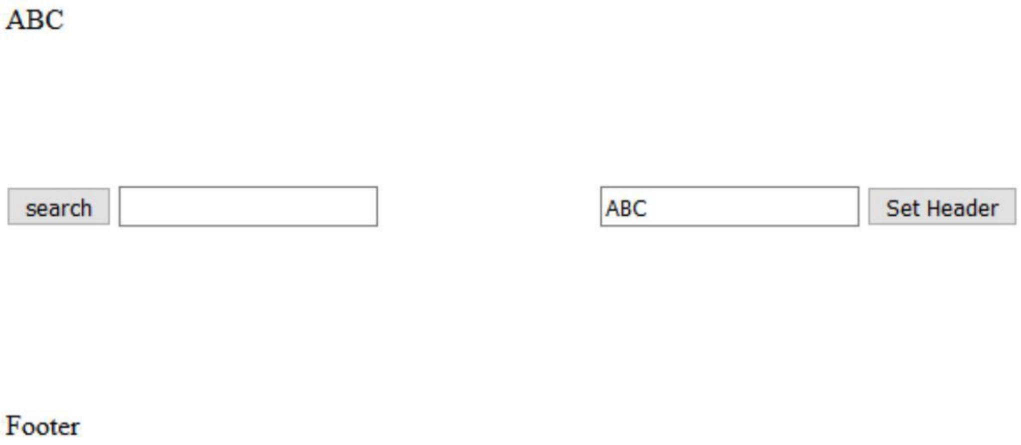
WebForm2.aspx

```
<%@ 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;
```

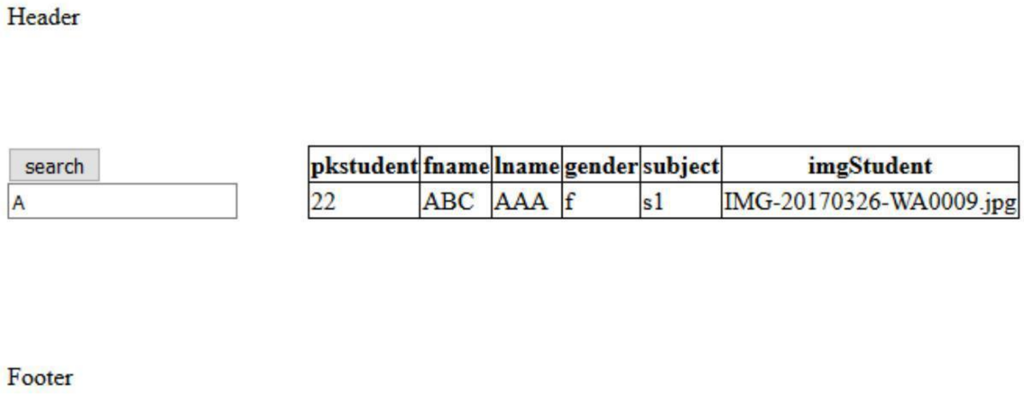
```
namespace masternew
{
    public partial class WebForm2 : System.Web.UI.Page
    {
        protected void Page_Init(object sender, EventArgs e)
        {
            ((admin)Master).Btnsave.Click += new
EventHandler(Btnsave_Click);
        }
        protected void Page_Load(object sender, EventArgs 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();
        }
    }
```



ABC

ABC

Footer



Header

A

pkstudent	fname	lname	gender	subject	imgStudent
22	ABC	AAA	f	s1	IMG-20170326-WA0009.jpg

Footer