



DOTNET LAB MANUAL

COMPUTER ENGINEERING



Name: Rupali Khunt

Sem:6th

Er.no.170473107013

Contents

PRACTICAL-1	1
AIM: INTRODUCTION TO C#:	1
PRACTICAL-2	10
AIM: GTU PROGRAMS:	10
PRACTICAL-3	16
AIM: OVERLOADING	16
PRACTICAL-4	22
AIM: REFLECTION	22
PRACTICAL-5	24
AIM: FILE HANDLING	24
PRACTICAL-6	30
AIM: WINDOWS FORM APPLICATION	30
PRACTICAL-7	34
AIM: ASP.NET VALIDATION CONTROL	34
PRACTICAL-8	37
AIM: INTRODUCTION TO MASTER PAGES	37
PRACTICAL-9	44
AIM: WEB SERVICES	44

PRACTICAL-1

AIM: INTRODUCTION TO C#:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace Practical_1
{
    class Program
    {
        static int j = 90;
        public enum TimeOfDay
        {
            Morning = 0,
            Afternoon = 1,
            Evening = 2
        }
        public static void Main(string[] args)
        {
            Console.WriteLine("First Program");

            //1.0 Variables
            // datatype identifier
            int i; //This statement declares an int named i.
```

```
        //The compiler won't actually let us use this
variable until we

        //have initialized it with a value, but the
declaration allocates

        i = 25;
        Console.WriteLine("Scope of Variables.\n1:");
        int j;
        for (/*int*/ j = 0; j < 2; j++) //removing comment from
for loop will raise error
        {
            Console.Write("{0} {1}\n", j, Program.j);
        }
        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");

        //As the name implies, a constant is a variable whose
value cannot be changed throughout its lifetime:

        // Characteristics:
```

//1. They must be initialized when they are declared, and once a value has been assigned, it can never be overwritten.

```
const int valConst = 100; // This value cannot be changed.
Console.WriteLine("{0} is constant value", valConst);
```

```
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);

Console.WriteLine("\nInteger Types");
sbyte sb = 33;
short s = 33;
int _i = 33;
long l = 33L;
//Unsigned Integers
byte b = 33;
ushort us = 33;
uint ui = 33U;
ulong ul = 33UL;
Console.WriteLine("{0} {1} {2} {3} {4} {5} {6} {7}", sb,
s, _i, l, b, us, ui, ul);
//Floating point types
float f = 11.22334455F;
double d = 11.2233445566778899;
Console.Write("\nFloat and Double:\n");
Console.WriteLine("{0} and {1}", f, d);
//Decimal Type
decimal dec = 111.222333444555666777888999M;
Console.WriteLine("Decimal:\n{0}", dec);
//Boolean
Console.WriteLine("\nBoolean:");
bool valBoolean = true;
Console.WriteLine("Status: " + valBoolean);
//Character
```

```
Console.WriteLine("\nCharacter:\nSingle Quote \'");
Console.WriteLine("Double Quote '\"");
Console.WriteLine("Back Slash \\");
char charA = 'A';
Console.WriteLine(charA);
charA = '\\0';
Console.WriteLine("Now null: " + charA);
//Console.WriteLine("\a"); //Notification Sound
//Thread.Sleep(1000);
//Console.Beep(); //another notification sound

//Predefined Reference Types
//object:
//We can use an object reference to bind to an object of
any particular sub-type.
//The object type implements a number of basic, general-
purpose methods, which include Equals(), GetHashCode(), GetType(), and
ToString().

object o1 = "Hi, I am an Object";
object o2 = 34;
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
string s1, s2;
```



```
s1 = "String 1";
s2 = s1;
Console.WriteLine("S1 is: {0} and s2 is {1}", s1, s2);
s2 = "New String 1";
Console.WriteLine("S1 is: {0} and s2 is {1}", s1, s2);
s1 = "c:\\NewFolder\\Hello\\P1.cs";
Console.WriteLine(s1);
s1 = @"c:\NewFolder\Hello\P1.cs";
Console.WriteLine(s1);
s1 = @"We can also write
like this";
Console.WriteLine(s1);
```

```
//Flow Control
//The if Statement
bool isZero;
Console.WriteLine("\nFlow Control: (if)\ni is " + i);
if (i == 0)
{
    isZero = true;
    Console.WriteLine("i is Zero");
}
else
{
    isZero = false;
    Console.WriteLine("i is Non - zero");
}
//Switch
```

```
int integerA = 2;
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]);
}
static 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;
    }
}

public class Vector
{
    public int value;
}
}
```

```
111.222333444555666777888999

Boolean:
Status: True

Character:
Single Quote '
Double Quote "
Back Slash \
A
Now null:
Hi, I am an Object
-1198555787 System.String
34 System.Int32
False
S1 is: String 1 and s2 is String 1
S1 is: String 1 and s2 is New String 1
c:\NewFolder\Hello\P1.cs
c:\NewFolder\Hello\P1.cs
We can also write
like this

Flow Control: (if)
i is 25
i is Non - zero

Switch:
integerA = 2
Good morning!
```

PRACTICAL-2

AIM: GTU PROGRAMS:

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 Practical_2
{
    class Program
    {
        static void Main(string[] args)
        {
            for(int i=5;i>0;i--)
            {
                for(int j=0;j<i;j++)
                {
                    Console.Write("@");
                }
                Console.WriteLine(" ");
            }
            Console.ReadKey();
        }
    }
}
```



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

1

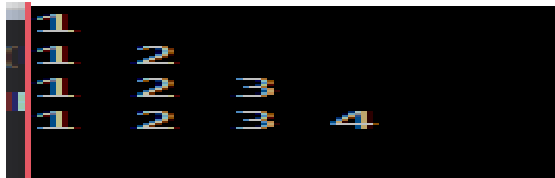
1 2

1 2 3

1 2 3 4

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Practical_2
{
    class pattern1
    {
        static void Main(string[] args)
        {
            int i, j;
            for (i = 1; i < 5; i++)
            {
                for (j = 1; j <= i; j++)
                {
                    Console.Write(j + " ");
                }
                Console.WriteLine();
            }
            Console.ReadKey();
        }
    }
}
```

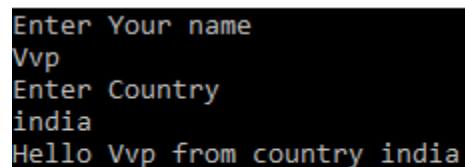


3) Write C# code to prompt a user to input his/her name and country name and then the output will be shown as an example below:

Hello Ram from country India.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Practical_2
{
    class stringprint
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter Your name");
            string name = Console.ReadLine();
            Console.WriteLine("Enter Country");
            string country = Console.ReadLine();
            Console.WriteLine("Hello " + name + " from country " +
country);
            Console.ReadLine();
        }
    }
}
```



```
Enter Your name
Vvp
Enter 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;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Practical_2
{
    class inheritance
    {
        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);
            Console.ReadLine();
        }
    }
}

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";
    }
}
}
```

```
Details Car 1: Swift -Maruti and Have AGS
Car 2: XUV500 -Mahindra
```

PRACTICAL-3

AIM: OVERLOADING

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

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace practical_3
{
    class method_over
    {
        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)
    {
        method_over p = new method_over();
        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;

}

```

```
Value of a:
20
Value of b:
30
Addition is:50
Input the size of the square matrix: 2
Input elements in the first matrix :
0,0:1
0,1:2
1,0:3
1,1:4
Input elements in the Second matrix :
0,0:4
0,1:3
1,0:2
1,1:1

The First matrix is :

1      2
3      4
The Second matrix is :

4      3
2      1
Addition of Two Matrix:

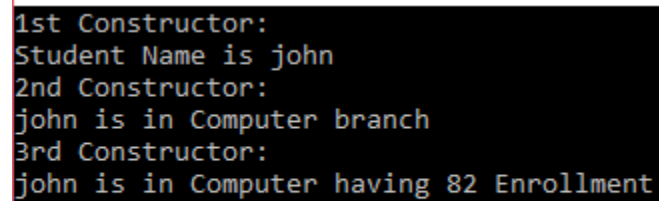
5      5
5      5      Enter 1st vector
X:
5
Y:
3
Z:
1
Enter 2nd vector
X:
2
Y:
4
Z:
6
Addition of Two vectors is:
<7.7.7>
```

Write a c# program that create student object. Overload constructor 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 practical_3
{
    class const_over
    {
        public int ID { get; set; }
        public string Name { get; set; }
        String name, branch;
        int enroll;
        const_over(String Stname)
        {
            name = Stname;
            Console.WriteLine("1st Constructor:");
            Console.WriteLine("Student Name is " + Stname);
        }
        const_over(String Stname, String Stbranch)
        {
            name = Stname;
            branch = Stbranch;
            Console.WriteLine("2nd Constructor:");
            Console.WriteLine(Stname + " is in " + Stbranch + "
branch");
        }
        const_over(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)
    {
        const_over p = new const_over("john");
        const_over p1 = new const_over("john", "Computer");
        const_over p2 = new const_over("john", "Computer", 82);
        Console.ReadLine();
    }
}
```



```
1st Constructor:
Student Name is john
2nd Constructor:
john is in Computer branch
3rd Constructor:
john is in Computer having 82 Enrollment
```


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.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Reflection;
namespace Practical_4
{
    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);
        }
    }
    class Program
    {
        public static void Main(String[] a)
        {
            //Type T =    .GetType();
```

```

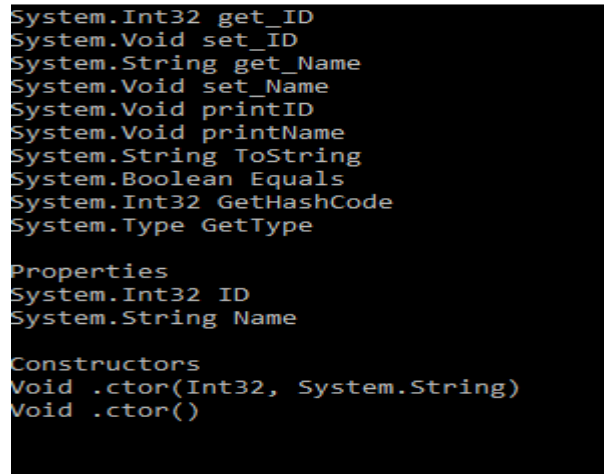
    Type T = (typeof(Customer));

    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());
    }
    Console.ReadKey();
}
}
}

```



```

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

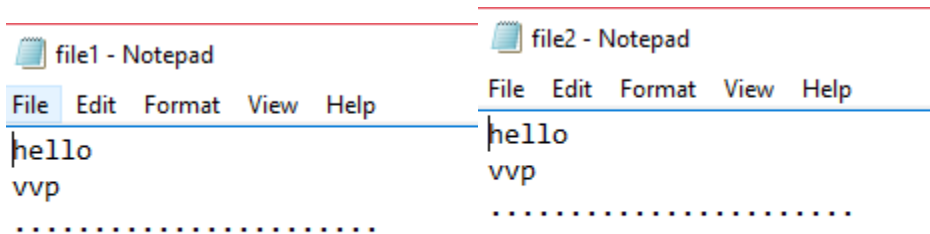
AIM: FILE HANDLING

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 copyfile
    {
        static void Main(string[] args)
        {
            CopyFile cp = new CopyFile();
            String file1 = @"D:\study\file1.txt";
            String file2 = @"D:\study\file2.txt";
            cp.copyFile(file1, file2);
            Console.ReadKey();
        }
    }
    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);
    }
}
}
```

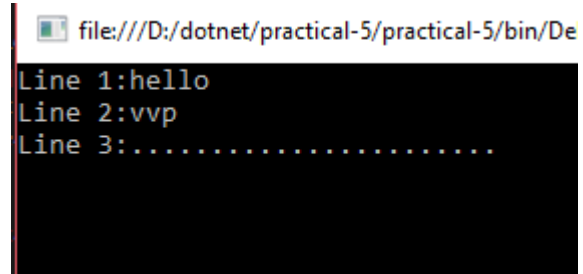


Write a C# Program to Read Lines from a File until the End of File is Reached.

```
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:\study\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();
            }
        }
    }
}
```

```
    }  
}  
}
```



file:///D:/dotnet/practical-5/practical-5/bin/De

```
Line 1:hello  
Line 2:vvp  
Line 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[] Directories =
Directory.GetDirectories(@"D:\study");
            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:\study");
            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

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.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.IO;

namespace StudentReistration
{
    public partial class Form1 : Form
    {
        string imgPath;
        public Form1()
        {
            InitializeComponent();

            private void radioButton2_CheckedChanged(object sender,
EventArgs e)
            {

            private void btnImage_Click(object sender, EventArgs e)
            {
                openFileDialog1.Filter = "Jpg|*.jpg";
                if (openFileDialog1.ShowDialog() == DialogResult.OK)
                {
```

```

        imgPath = @"C:\Users\CRP\Desktop\Images\"+
openFileDialog1.SafeFileName;
        imgStudent.Image =
Image.FromFile(openFileDialog1.FileName);
        //MessageBox.Show(imgPath);
    }
}

private void btnCancel_Click(object sender, EventArgs e)
{
    Environment.Exit(0);
}

private void btnSave_Click(object sender, EventArgs e)
{
    string source = @"Data Source=crp-pc\mydatabase;Initial
Catalog=temp1;Integrated Security=True";
    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 pkStudent = i + 1;

    string insert = "insert into tblStudent (pkStudent,
fName,dob, imgStudent) values (
"+pkStudent+", '"+txtFname.Text+"', '"+dateDob.Value.Date +"' ,'" +
(imgPath==null?"":imgPath) +"' )";
    cmd = new SqlCommand(insert,conn);

    i = cmd.ExecuteNonQuery();
    if(imgPath!=null)
        imgStudent.Image.Save(imgPath);
    MessageBox.Show("You are Done!!!");
    InitializeComponent();
}

}
}

```

PROGRAM.CS

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

namespace StudentReistration
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

Student Registration


Personal Details

Name:

Gender: ☒ Male ☐ Female

Mobile: Email:

DoB: ,



Name	<input type="text" value="vvp"/>	
Mobile	<input type="text" value="66494"/>	*
Password	<input type="text" value="746587"/>	
Confirm Password	<input type="text" value="746587"/>	
Semester	<input type="text" value="2"/>	RangeValidator

- Must Enter Number 10 digits Long
- 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
runat="serve      ID="ContentPlaceHolder1"
r">
                        <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>
```

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\squlexpress;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

search	<input type="text"/>	ABC	Set Header
--------	----------------------	-----	------------

Footer

Header

search

A

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

Footer

PRACTICAL-9

AIM: WEB SERVICES

WebForm1.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WebForm1.aspx.cs" Inherits="WebServices.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>
    <form id="form1" runat="server">
        <asp:Label ID="lbln2" runat="server" Text="No.1"></asp:Label>
        <asp:TextBox ID="txtno1" runat="server">
        </asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator1"
runat="server"
        ControlToValidate="txtno1" ErrorMessage="value must be
required"></asp:RequiredFieldValidator>
        <br />
        <asp:Label ID="lbln1" runat="server" Text="No.2"></asp:Label>
        <asp:TextBox ID="txtno2" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator2"
runat="server"
        ControlToValidate="txtno2" ErrorMessage="value must be
required"></asp:RequiredFieldValidator>
        <br />
        <asp:Button ID="btnadd" runat="server" Text="add"
onclick="btnadd_Click" />
        <asp:Button ID="btnsub" runat="server" onclick="btnsub_Click"
Text="Sub" />
        <asp:Button ID="btnmul" runat="server" onclick="btnmul_Click"
style="width: 35px" Text="mul" />
        <asp:Button ID="btndiv" runat="server" onclick="btndiv_Click"
Text="Div" />
        <asp:Label ID="lblresult" runat="server" Text="Label"></asp:Label>
    </form>
</body>
</html>
```

Webfrom1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebServices
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        WebService1 calc = new WebService1();
        protected void btnadd_Click(object sender, EventArgs e)
        {
            lblresult.Text = calc.add(Convert.ToInt16(txtno1.Text),
            Convert.ToInt16(txtno2.Text)).ToString();
        }

        protected void btnsub_Click(object sender, EventArgs e)
        {
            lblresult.Text = calc.sub(Convert.ToInt16(txtno1.Text),
            Convert.ToInt16(txtno2.Text)).ToString();
        }

        protected void btnmul_Click(object sender, EventArgs e)
        {
            lblresult.Text = calc.mul(Convert.ToInt16(txtno1.Text),
            Convert.ToInt16(txtno2.Text)).ToString();
        }

        protected void btndiv_Click(object sender, EventArgs e)
        {
            lblresult.Text = calc.div(Convert.ToInt16(txtno1.Text),
            Convert.ToInt16(txtno2.Text)).ToString() ;
        }

        protected void btncal_Click(object sender, EventArgs e)
        {
        }
    }
}
```


Webservicecs.asmx

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

namespace WebServices
{
    /// <summary>
    /// Summary description for WebService1
    /// </summary>
    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.ComponentModel.ToolboxItem(false)]
    // To allow this Web Service to be called from script, using
    ASP.NET AJAX, uncomment the following line.
    // [System.Web.Script.Services.ScriptService]
    public class WebService1 : System.Web.Services.WebService
    {

        [WebMethod]
        public string HelloWorld()
        {
            return "Hello World";
        }
        [WebMethod]
        public int add(int a,int b)
        {
            return a + b;
        }
        [WebMethod]
        public int sub(int a,int b)
        {
            return a-b;
        }
        [WebMethod]
        public int mul(int a,int b)
        {
            return a*b;
        }
    }
}
```

```
[WebMethod]
public int div(int a,int b)
{
    return a/b;
}

}
```

No1	<input type="text" value="2"/>			
No2	<input type="text" value="4"/>			
<input type="button" value="add"/>	<input type="button" value="sub"/>	<input type="button" value="mul"/>	<input type="button" value="div"/>	result

WebService1

The following operations are supported. For a formal definition, please review the [Service Description](#).

- [add](#)
- [div](#)
- [mul](#)
- [sub](#)

This web service is using <http://tempuri.org/> as its default namespace.

Recommendation: Change the default namespace before the XML Web service is mad

Each XML Web service needs a unique namespace in order for client applications to distinguish it from ot Web services should use a more permanent namespace.

Your XML Web service should be identified by a namespace that you control. For example, you can use y they need not point to actual resources on the Web. (XML Web service namespaces are URIs.)