

# .NET TECHNOLOGY

Lab Manual

RADHIKA KANABAR

## Contents

Introduction to C# .....	1
GTUPrograms .....	9
Overloading.....	16
Reflection API .....	22
Perform File Handling. ....	25
Windows Form Application.....	29
ASP.NET Validation Control .....	32
Introduction to Master Pages. ....	34

## Practical 1

AIM:

### Introduction to C#

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Program1
{
    class vector
    {
        public int value;
    }

    class Program1
    {
        static int i = 25;

        public enum TimeOfDay
        {
            Morning = 0,
            Afternoon = 1,
            Evening = 2
        }

        static void Main(string[] args)
        {
            Console.WriteLine("This is first program");

            //Scope of variables

            int i=5;
```

```
Console.WriteLine("Scope of the variable {0}",i);
for (i = 0; i < 2; i++)
{
    Console.WriteLine("{0} {1}",i,Program1.i);
}
for (int k = 0; k < 2; k++)
{
    Console.WriteLine("{0}",k);
}
//Constant
const int valueConst=25;
Console.WriteLine("{0}",valueConst);
//valueConst = 15;
const int valueConst2 = 15;
Console.WriteLine("{0}", valueConst2);
//valueConst = valueConst2;
Console.WriteLine("{0}",valueConst);
//Value Type DataTypes
Console.WriteLine("Value Type");
int val1, val2;
val1 = 50;
Console.WriteLine("val1= {0}",val1);
val2 = val1;
Console.WriteLine("val1= {0} val2= {1}", val1,val2);
//Reference Type
Console.WriteLine("Reference Type");
vector x, y;
x = new vector();
x.value = 15;
y = x;
```

```
Console.WriteLine("x = {0}  y = {1}", x.value,y.value);

y.value = 151;

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

Console.WriteLine("\n Interger Types");

sbyte sb = 22;

short s = 22;

int i1 = 22;

long l = 22L;

Console.WriteLine("{0} sbtye\n{1} short\n{2} int\n{3}
long\n",sb,s,i1,l);

Console.WriteLine("Unsigned Integers");

byte b = 21;

ushort us = 21;

uint ui = 21;

ulong ul = 21;

Console.WriteLine("{0} btye\n{1} ushort\n{2} uint\n{3} ulong\n", b,
us, ui, ul);

Console.WriteLine("Floating Point");

float f = 11.22334455F;

double d = 11.2233445566778899;

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

decimal dec = 111.2223334444555666777888999M;

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

Console.WriteLine("\nBoolean:");

bool valBoolean = true;

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

Console.WriteLine("\nCharacter:\nSingle Quote \'');

Console.WriteLine("Double Quote '\"");

Console.WriteLine("Back Slash \\");

char charA = 'A';

Console.WriteLine(charA);
```

```
int integerA = 2;

Console.WriteLine("Predefined Reference Type");

Object o1 = "This is object 1";

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 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);

Console.WriteLine("Flow control 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");
}

//else if
Console.WriteLine("\nType in a string:");
string input;
input = Console.ReadLine();
if (input == "")
{
    Console.WriteLine("You typed in an empty string");
}
else if (input.Length < 5)
{
    Console.WriteLine("The string had less than 5 characters");
}
else if (input.Length < 10)
{
    Console.WriteLine("The string had at least 5 but less than 10
characters");
}
Console.WriteLine("The string was " + input);
Console.WriteLine("\nSwitch:");

switch (integerA)
{
    case 1:
        Console.WriteLine("integerA = 1");
        break;
```

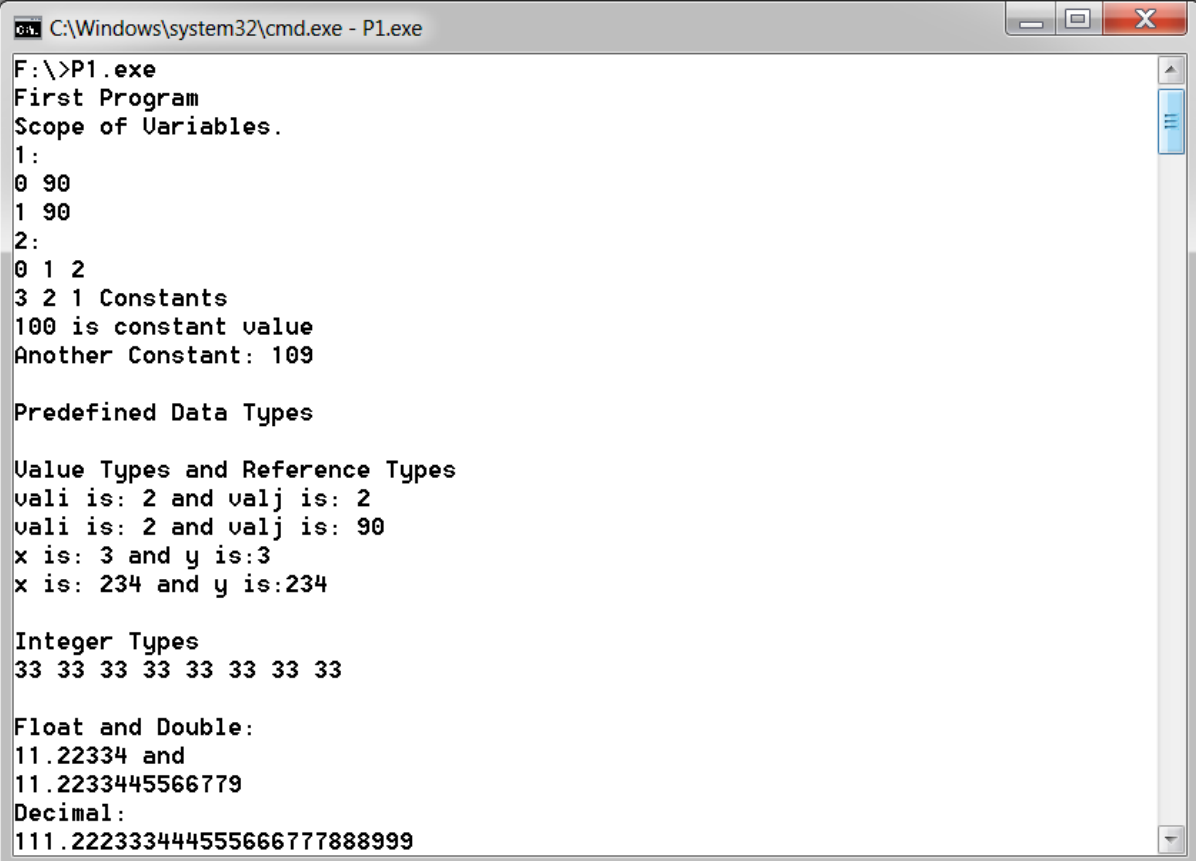
```
        case 2:
            Console.WriteLine("integerA = 2");
            //goto case 3;
            break;
        case 3:
            Console.WriteLine("integerA = 3");
            break;
        default:
            Console.WriteLine("integerA is not 1, 2, or 3");
            break;
    }

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

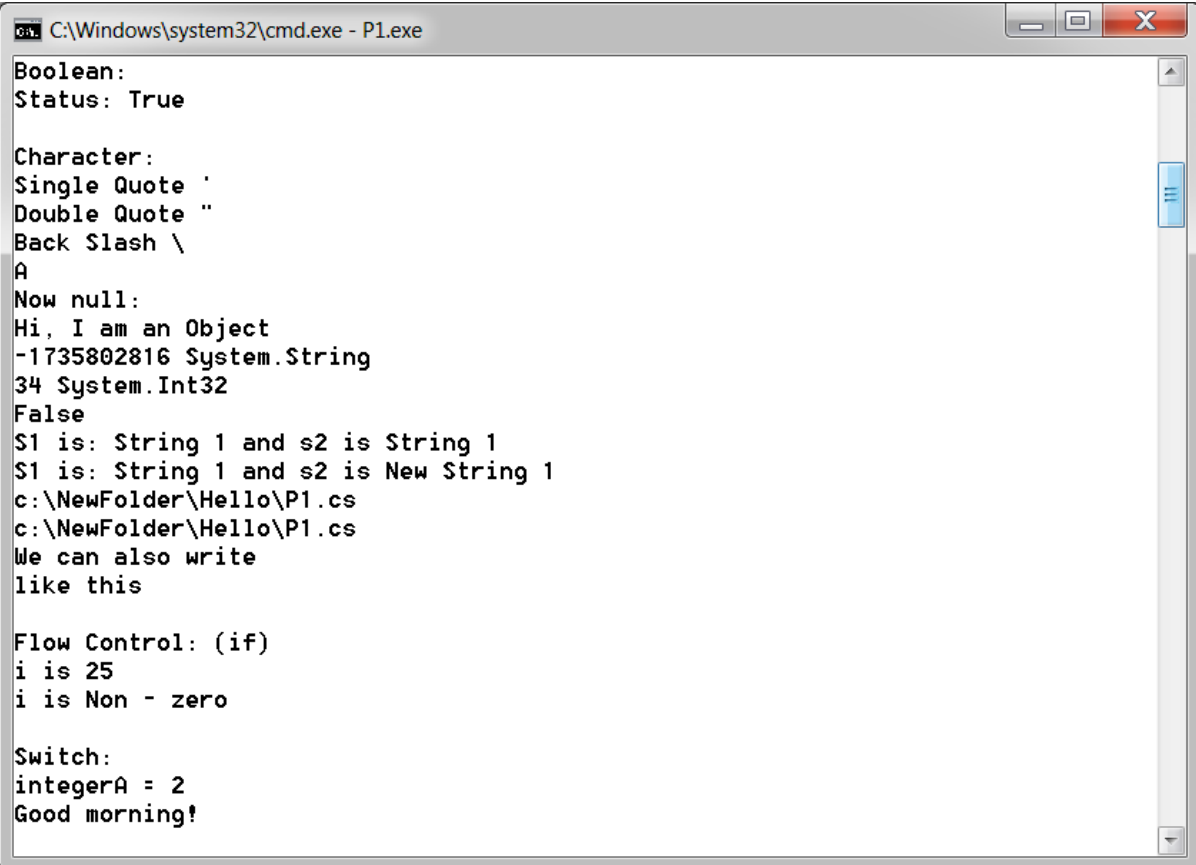
static void WriteGreeting(TimeOfDay timeOfDay)
{
    switch (timeOfDay)
    {
        case TimeOfDay.Morning:
            Console.WriteLine("Good morning!");
            break;
        case TimeOfDay.Afternoon:
            Console.WriteLine("Good afternoon!");
            break;
        case TimeOfDay.Evening:
            Console.WriteLine("Good evening!");
            break;
        default:
    }
```



```
        Console.WriteLine("Hello!");  
        break;  
    }  
}  
}
```



```
C:\Windows\system32\cmd.exe - P1.exe  
F:\>P1.exe  
First Program  
Scope of Variables.  
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
```



```
C:\Windows\system32\cmd.exe - P1.exe
Boolean:
Status: True

Character:
Single Quote '
Double Quote "
Back Slash \
A
Now null:
Hi, I am an Object
-1735802816 System.String
34 System.Int32
False
$1 is: String 1 and s2 is String 1
$1 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:

GTUPrograms

Program 1:

AIM: 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;
```

```
namespace Pattern1
```

```
{
```

```
    class Program
```

```
    {
```

```
        static void Main(string[] args)
```

```
        {
```

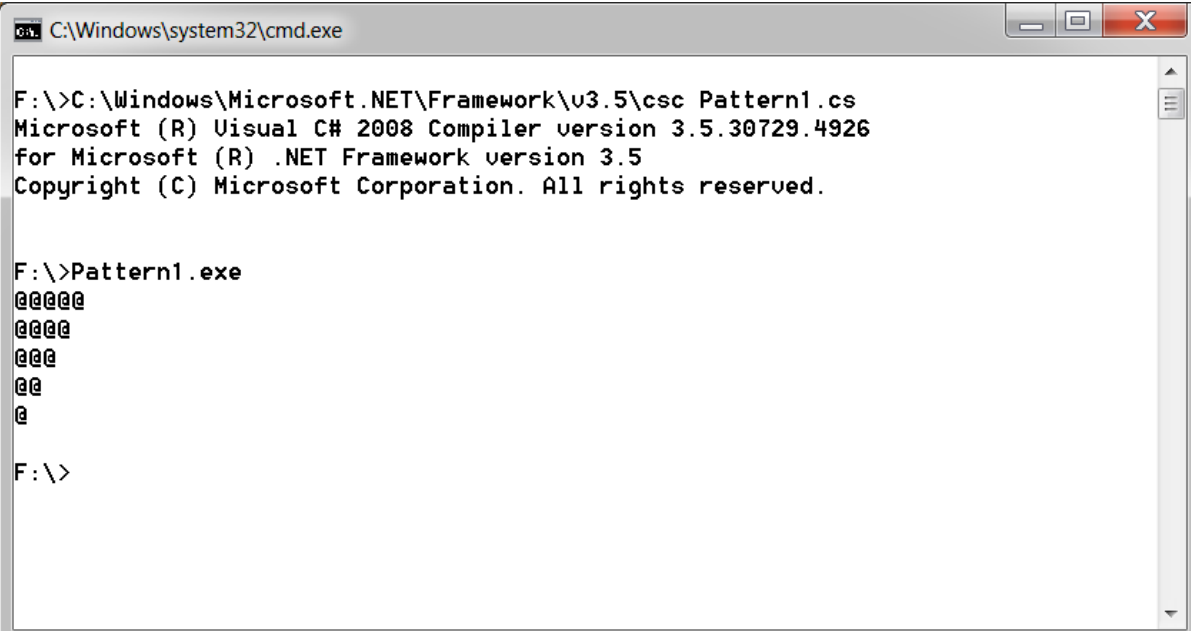
```
            for (int i = 5; i > 0; i--)
```

```
            {
```

```
                for (int j = 0; j < i; j++)
```

```
                {
```

```
        Console.Write("@");  
    }  
    Console.WriteLine();  
}  
Console.ReadKey();  
}  
}
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The command prompt displays the following text:

```
F:\>C:\Windows\Microsoft.NET\Framework\v3.5\csc Pattern1.cs  
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.4926  
for Microsoft (R) .NET Framework version 3.5  
Copyright (C) Microsoft Corporation. All rights reserved.  
  
F:\>Pattern1.exe  
@@@@@  
@@@@@  
@@@  
@@  
@  
F:\>
```

## Program 2

AIM: 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

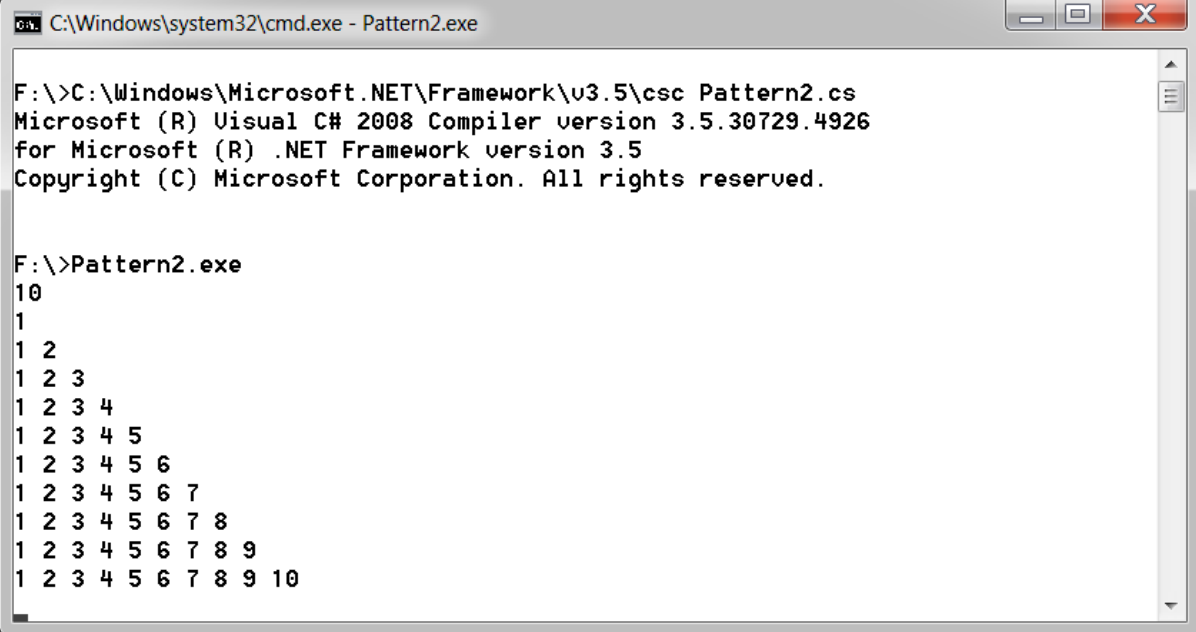
```
using System;
```

```
using System.Collections.Generic;
```

```
using System.Linq;

using System.Text;

namespace Pattern2
{
    class Program
    {
        static void Main(string[] args)
        {
            String s = Console.ReadLine();
            int value = int.Parse(s);
            for (int i = 1; i <= value; i++)
            {
                for (int j = 1; j <=i; j++)
                {
                    Console.Write("{0} ",j);
                }
                Console.WriteLine();
            }
            Console.ReadKey();
        }
    }
}
```



```
C:\Windows\system32\cmd.exe - Pattern2.exe

F:\>C:\Windows\Microsoft.NET\Framework\v3.5\csc Pattern2.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.4926
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

F:\>Pattern2.exe
10
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9
1 2 3 4 5 6 7 8 9 10
```

### Program 3

AIM: 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;

namespace PrintNameCountry
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Enter name");

            String name = Console.ReadLine();

            Console.WriteLine("Enter Country");


            String country = Console.ReadLine();
```

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

        Console.ReadKey();

    }

}
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The command prompt displays the following text:

```
F:\>C:\Windows\Microsoft.NET\Framework\v3.5\csc Country.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.4926
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

F:\>Country.exe
Enter name
Parth
Enter Country
India
Hello Parth from country India
F:\>
```

#### Program 4

AIM: 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;

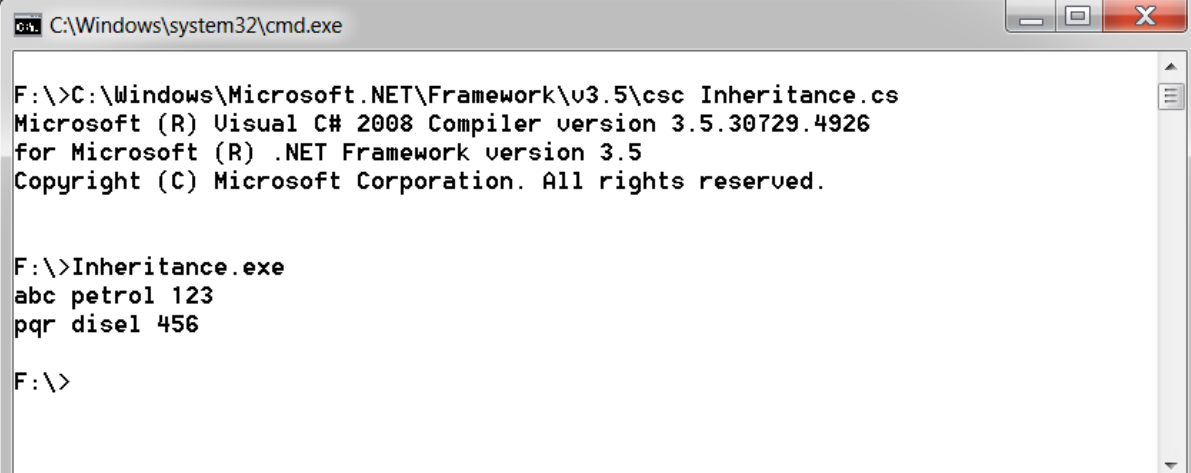
using System.Text;

namespace Inheritance
{
    class Car
    {
        protected String name, fuel,id;

    }
```

```
class Maruti: Car
{
    internal Maruti(String name, String fuel, String id)
    {
        this.name = name;
        this.fuel = fuel;
        this.id = id;
        Console.WriteLine("{0} {1} {2}",this.name, this.fuel, this.id);
    }
}
class Mahindra : Car
{
    internal Mahindra(String name, String fuel, String id)
    {
        this.name = name;
        this.fuel = fuel;
        this.id = id;
        Console.WriteLine("{0} {1} {2}",this.name, this.fuel, this.id);
    }
}
class Program
{
    static void Main(string[] args)
    {
        Maruti obj1= new Maruti("abc","petrol","123");
        Mahindra obj2 =new Mahindra("pqr","disel","456");
        Console.ReadKey();
    }
}
```





A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\system32\cmd.exe'. The command prompt shows the following sequence of commands and output:

```
F:\>C:\Windows\Microsoft.NET\Framework\v3.5\csc Inheritance.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.4926
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

F:\>Inheritance.exe
abc petrol 123
pqr diesel 456

F:\>
```

## Practical 3

AIM:

Overloading

1. 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;

namespace MethodOverloading
{
    class Vector
    {
        internal int x, y, z;

        internal Vector(int x, int y, int z)
        {
            this.x = x;
            this.y = y;
            this.z = z;
        }

        internal Vector() { }
    }

    class Matrix
    {
        internal int [,] m = new int[2,2];

        internal Matrix(){}

    }
```

```
class Program
{
    static void add(int a, int b)
    {
        int temp = a + b;
        Console.WriteLine(temp);
    }
    static void add(Vector a, Vector b)
    {
        Vector temp = new Vector();
        temp.x = a.x + b.x;
        temp.y = a.y + b.y;
        temp.z = a.z + b.z;
        Console.WriteLine("{0}x {1}y {2}z", temp.x, temp.y, temp.z);
    }
    static void add(Matrix a, Matrix b)
    {
        Matrix temp = new Matrix();
        for (int i = 0; i < 2; i++)
        {
            for (int j = 0; j < 2; j++)
            {
                temp.m[i, j] = a.m[i, j] + b.m[i, j];
                Console.Write(temp.m[i, j] + "\t");
            }
            Console.Write("\n");
            Console.WriteLine();
        }
    }
}
```

```
static void Main(string[] args)
{
    Console.WriteLine("Enter Vector");

    Vector a = new Vector(int.Parse(Console.ReadLine()),
int.Parse(Console.ReadLine()), int.Parse(Console.ReadLine()));

    Vector b = new Vector(int.Parse(Console.ReadLine()),
int.Parse(Console.ReadLine()), int.Parse(Console.ReadLine()));

    add(a, b);

    Console.WriteLine("Enter integer");

    int x = int.Parse(Console.ReadLine());

    int y = int.Parse(Console.ReadLine());

    add(x, y);

    Console.WriteLine("Sum of Matrix is\n");

    Matrix m1 = new Matrix();

    Matrix m2 = new Matrix();

    m1.m[0, 0] = 2;

    m1.m[0, 1] = 2;

    m1.m[1, 0] = 2;

    m1.m[1, 1] = 2;

    m2.m[0, 0] = 3;

    m2.m[0, 1] = 3;

    m2.m[1, 0] = 3;

    m2.m[1, 1] = 3;

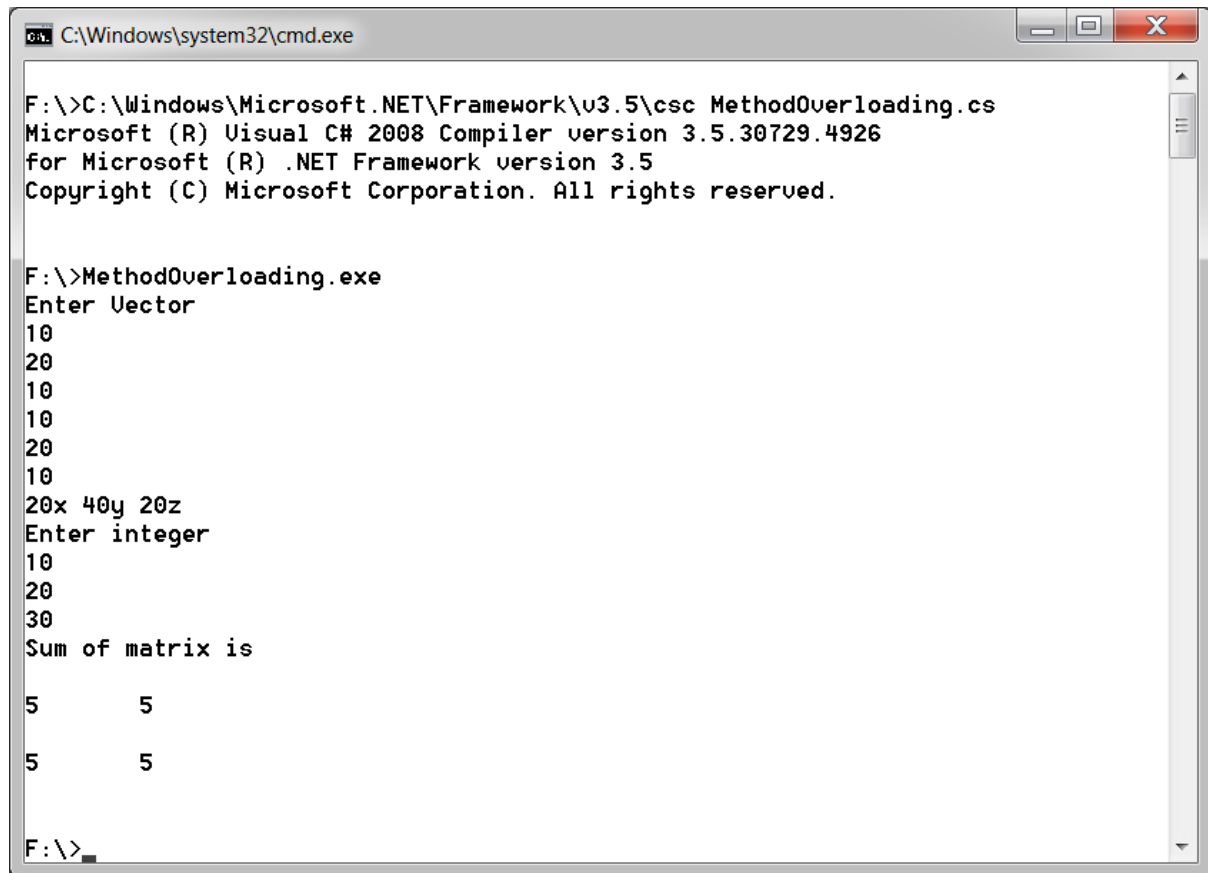
    add(m1, m2);

    Console.ReadKey();

}

}

}
```



```
C:\Windows\system32\cmd.exe

F:\>C:\Windows\Microsoft.NET\Framework\v3.5\csc MethodOverloading.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.4926
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

F:\>MethodOverloading.exe
Enter Vector
10
20
10
10
20
10
20x 40y 20z
Enter integer
10
20
30
Sum of matrix is

5      5

5      5

F:\>
```

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;
```

```
namespace ConstructorOverloading
```

```
{
```

```
    class Student
```

```
    {
```

```
String name,enroll_no,branch;

public Student(String name)
{
    this.name = name;
}

public Student(String name, String enroll_no)
{
    this.name = name;
    this.enroll_no = enroll_no;
}

public Student(String name, String enroll_no, String branch)
{
    this.name = name;
    this.enroll_no = enroll_no;
    this.branch = branch;
}

internal String getName()
{
    return this.name;
}

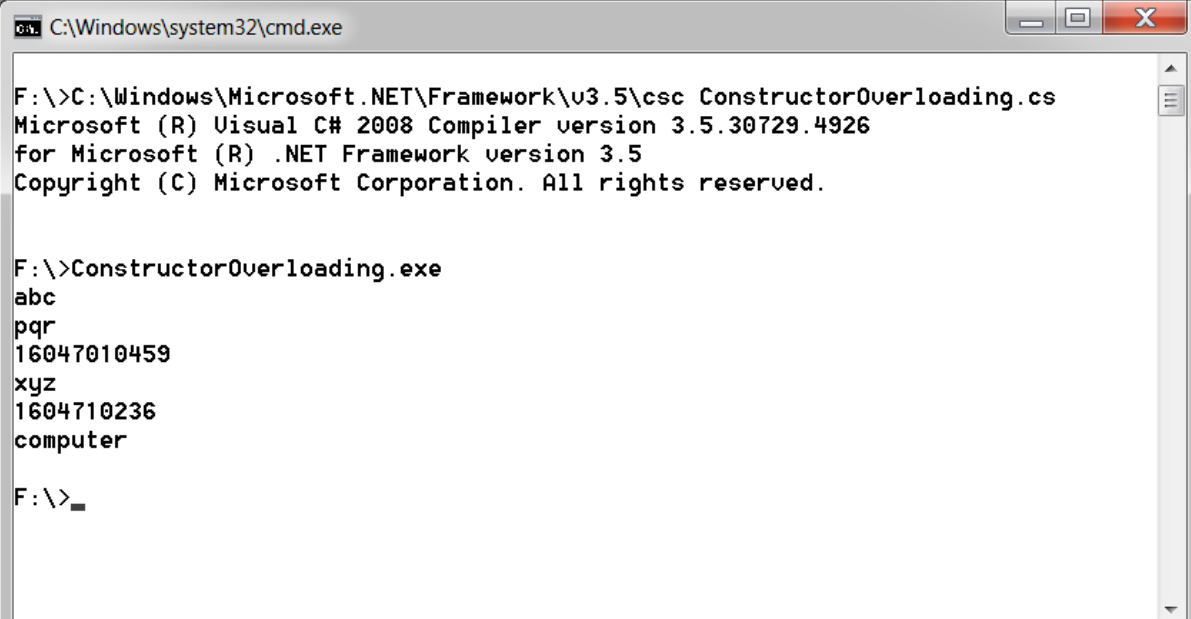
internal String getEnroll()
{
    return this.enroll_no;
}

internal String getBranch()
{
    return this.branch;
}

}

class Program
```

```
{  
    static void Main(string[] args)  
    {  
        Student s1 = new Student("abc");  
        Console.WriteLine(s1.getName());  
        Student s2 = new Student("pqr", "16047010459");  
        Console.WriteLine(s2.getName());  
        Console.WriteLine(s2.getEnroll());  
        Student s3 = new Student("xyz", "1604710236", "computer");  
        Console.WriteLine(s3.getName());  
        Console.WriteLine(s3.getEnroll());  
        Console.WriteLine(s3.getBranch());  
        Console.ReadKey();  
    }  
}
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The command prompt displays the following text:

```
F:\>C:\Windows\Microsoft.NET\Framework\v3.5\csc ConstructorOverloading.cs  
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.4926  
for Microsoft (R) .NET Framework version 3.5  
Copyright (C) Microsoft Corporation. All rights reserved.  
  
F:\>ConstructorOverloading.exe  
abc  
pqr  
16047010459  
xyz  
1604710236  
computer  
F:\>_
```

## Practical 4

AIM:

Reflection API

1. Create a c# program to find Methods, Properties and Constructors from class of running program.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Reflection;

namespace Reflection
{
    class Student
    {
        String name, enroll_no, branch;
        public Student(String name)
        {
            this.name = name;
        }
        public Student(String name, String enroll_no)
        {
            this.name = name;
            this.enroll_no = enroll_no;
        }
        public Student(String name, String enroll_no, String branch)
        {
            this.name = name;
```



```
        this.enroll_no = enroll_no;

        this.branch = branch;
    }

    public String getName()
    {
        return this.name;
    }

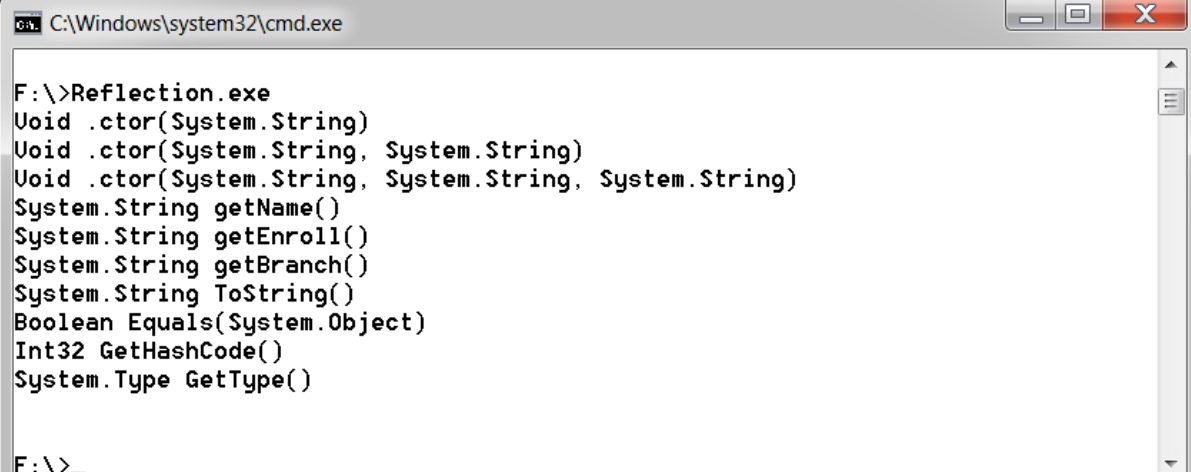
    public String getEnroll()
    {
        return this.enroll_no;
    }

    public String getBranch()
    {
        return this.branch;
    }
}

class Program
{
    static void Main(string[] args)
    {
        Type t = Type.GetType("Reflection.Student");
        ConstructorInfo[] ci = t.GetConstructors();
        MethodInfo[] mi = t.GetMethods();
        foreach (ConstructorInfo c in ci)
        {
            Console.WriteLine(c.ToString());
        }

        foreach (MethodInfo m in mi)
        {
            Console.WriteLine(m.ToString());
        }
    }
}
```

```
        }  
        Console.ReadLine();  
    }  
}
```



A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The window shows the output of the "Reflection.exe" application. The output lists several methods and their return types: "Void .ctor(System.String)", "Void .ctor(System.String, System.String)", "Void .ctor(System.String, System.String, System.String)", "System.String getName()", "System.String getEnroll()", "System.String getBranch()", "System.String ToString()", "Boolean Equals(System.Object)", "Int32 GetHashCode()", and "System.Type GetType()". The prompt "F:\>" is visible at the bottom.

```
C:\Windows\system32\cmd.exe  
F:\>Reflection.exe  
Void .ctor(System.String)  
Void .ctor(System.String, System.String)  
Void .ctor(System.String, System.String, System.String)  
System.String getName()  
System.String getEnroll()  
System.String getBranch()  
System.String ToString()  
Boolean Equals(System.Object)  
Int32 GetHashCode()  
System.Type GetType()  
F:\>
```

## Practical 5

AIM:

Perform File Handling.

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

### Program 1

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.IO;

namespace CopyFile1

{

    class Program

    {

        static void Main(string[] args)

        {

            String file1 = @"F:\file1.txt";

            String file2 = @"F:\file2.txt";

            using (StreamReader reader = new StreamReader(file1))

            {

                using (StreamWriter writer = new StreamWriter(file2))

                {

                    writer.Write(reader.ReadToEnd());

                }

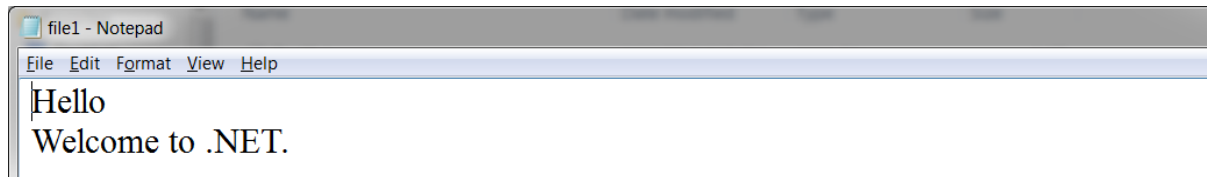
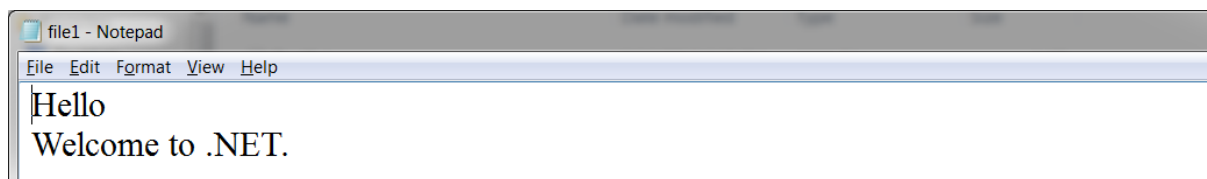
            }

        }

    }

}
```

```
}  
}
```

**FILE1:****FILE2:**

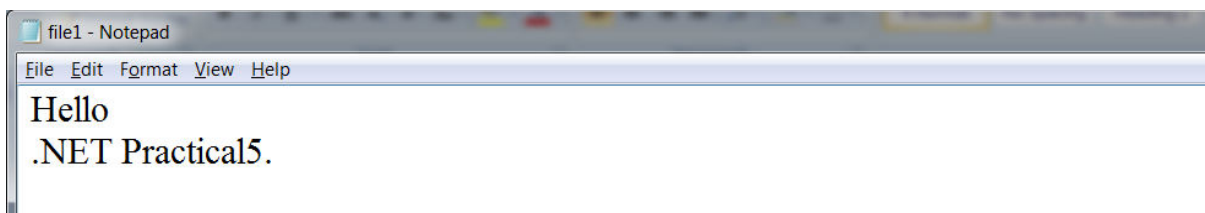
2. Write a C# Program to Read Lines from a File until the End of File is Reached.

**Program 2**

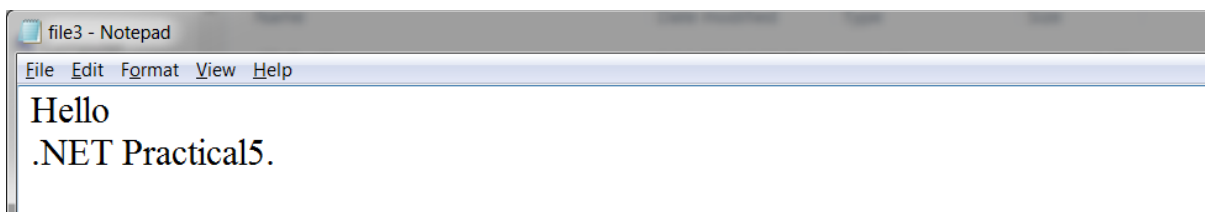
```
using System;  
using System.Collections.Generic;  
using System.Linq;  
using System.Text;  
using System.IO;  
namespace CopyFile2  
{  
    class Program  
    {  
        static void Main(string[] args)  
        {  
            String file1 = @"F:\file1.txt";  
            String file2 = @"F:\file2.txt";  
            String content = null;
```

```
using (StreamReader reader = new StreamReader(file1))
{
    using (StreamWriter writer = new StreamWriter(file2))
    {
        while ((content = reader.ReadLine()) != null)
        {
            writer.WriteLine(content);
        }
    }
}
```

FILE1:



FILE3:



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

Program 3

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

```
using System.IO;

namespace filepractical3
{
    class Program
    {
        static void Main(string[] args)
        {
            String[] Directories = Directory.GetDirectories(@"F:\DotNET");

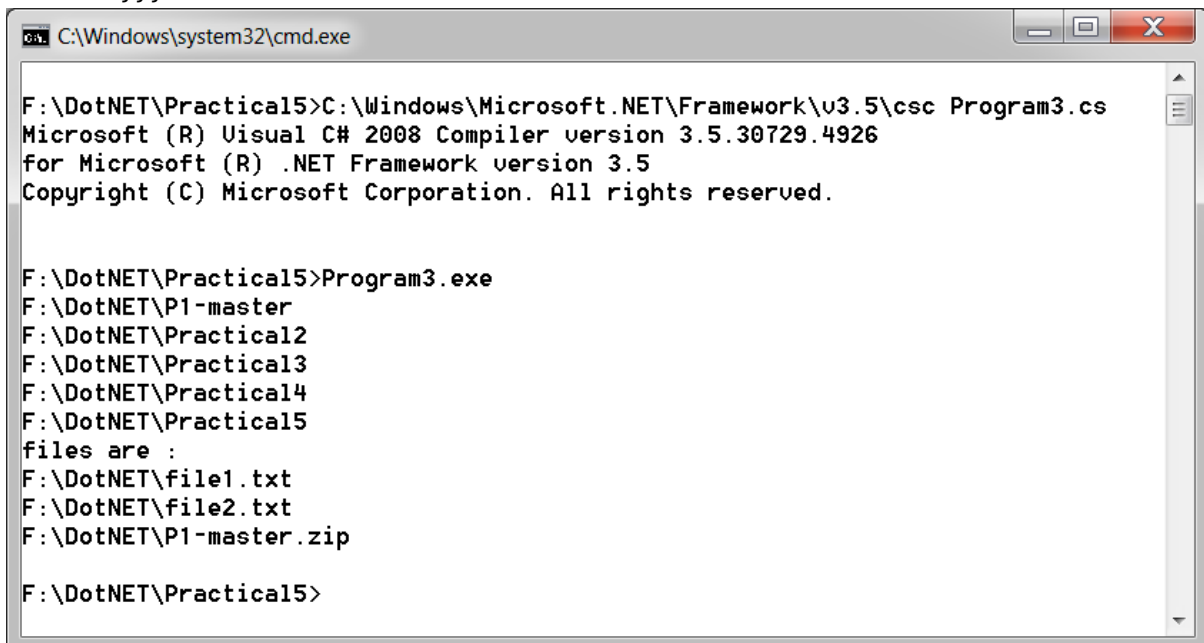
            foreach (string dir in Directories)
            {
                Console.WriteLine(dir);
            }

            Console.WriteLine("files are :");

            String[] files = Directory.GetFiles(@"F:\DotNET");

            foreach (string file in files)
            {
                Console.WriteLine(file);
            }

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



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe". The user is in the directory "F:\DotNET\Practical15". They first compile the program "Program3.cs" using the C# compiler "csc". The output shows the compiler version (3.5.30729.4926) and copyright information. Then, they run the program "Program3.exe". The output lists the directories found in "F:\DotNET": "P1-master", "Practical12", "Practical13", "Practical14", and "Practical15". It then lists the files found: "file1.txt", "file2.txt", and "P1-master.zip". The prompt returns to "F:\DotNET\Practical15>" after the program finishes.

```
C:\Windows\system32\cmd.exe
F:\DotNET\Practical15>C:\Windows\Microsoft.NET\Framework\v3.5\csc Program3.cs
Microsoft (R) Visual C# 2008 Compiler version 3.5.30729.4926
for Microsoft (R) .NET Framework version 3.5
Copyright (C) Microsoft Corporation. All rights reserved.

F:\DotNET\Practical15>Program3.exe
F:\DotNET\P1-master
F:\DotNET\Practical12
F:\DotNET\Practical13
F:\DotNET\Practical14
F:\DotNET\Practical15
files are :
F:\DotNET\file1.txt
F:\DotNET\file2.txt
F:\DotNET\P1-master.zip

F:\DotNET\Practical15>
```

## Practical 6

AIM:

Windows Form Application

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

namespace WindowsForm1
{
    public partial class Form1 : Form
    {
        string imgPath; public String gender;
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {
        }

        private void button1_Click(object sender, EventArgs e)
        {
            String source = @"Data Source=CE3COMP3\squlexpress;Initial
            Catalog=DBstudent;Integrated Security=True;Pooling=False";
            SqlConnection con = new SqlConnection(source);
            con.Open();
            String ins = "insert into Tbl1(fname,Middlename,Lname,gender,Date)
            values('"+fname.Text+"','"+ Middlename.Text+ "','"+ Lname.Text +
            "','"+ +gender+"','"+ dateTimePicker1.Value.Date +"'");
            SqlCommand sc = new SqlCommand(ins, con);

            int i=sc.ExecuteNonQuery();
            if (i > -1)
            {
                MessageBox.Show("Entered into database");
            }
        }
    }
}
```

```
private void button3_Click(object sender, EventArgs e)
{
    openFileDialog1.Filter = "Png|*.png";
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
    {
        imgPath = @"C:\Users\CRP\Desktop\Images\" +
            openFileDialog1.SafeFileName;
        pictureBox.Image = Image.FromFile(openFileDialog1.FileName);
    }
}

private void Male_CheckedChanged(object sender, EventArgs e)
{
    if (Male.Checked)
    {
        gender = "Male";
    }
    else
    {
        gender = "Female";
    }
}
}
```



The screenshot displays a Windows application window titled "StudentRegistrationForm". The window has a standard Windows title bar with minimize, maximize, and close buttons. The main content area is a light gray form with the following controls:

- Name:** Three text input fields, each containing the text "abc".
- Gender:** Two radio buttons labeled "Male" and "Female". The "Female" radio button is selected.
- Date:** A date picker control showing "Saturday , March 30, 2019".
- submit:** A button located at the bottom right of the form.

A small modal dialog box is open over the bottom right of the main form. It has a blue title bar with a red close button. The dialog contains the text "Entered into database" and an "OK" button.

## Practical 7

AIM:

### ASP.NET Validation Control

RequiredFieldValidator

CompareValidator

RegularExpressionValidator

CustomValidator

RangeValidator

ValidationSummary

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="ASPWebApplication1.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">
        <div>

            </div>
            name
            <asp:TextBox ID="Txtname" runat="server"></asp:TextBox>
            <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
            ControlToValidate="Txtname" ErrorMessage="name is required" ForeColor="Red"
            ToolTip="Please enter name">*</asp:RequiredFieldValidator>
            <br />
            &nbsp;
            <br />
            email<asp:TextBox ID="Txtemail" runat="server"
            ontextchanged="TextBox1_TextChanged"></asp:TextBox>
            <asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server"
            ControlToValidate="Txtemail" ErrorMessage="not valid email address"
            ForeColor="Red" ToolTip="enter valid email"
            ValidationExpression="\w+([-+.' ]\w+)*@\w+([-.]\w+)*\.\w+([-.
            .]\w+)*">*</asp:RegularExpressionValidator>
            <br />
            <br />
            phone no<asp:TextBox ID="Txtphone" runat="server"
            ontextchanged="Txtphone_TextChanged"></asp:TextBox>
            <asp:RegularExpressionValidator ID="RegularExpressionValidator2"
            runat="server" ControlToValidate="Txtphone" ErrorMessage="not valid phone
            no" ForeColor="Red" ToolTip="enter 10 digit mobile no"
            ValidationExpression="[0-9]{10}">*</asp:RegularExpressionValidator>
            <br />
            <br />
```

```
password<asp:TextBox ID="Txtpassword" runat="server"></asp:TextBox>
<br />
<br />
confirm password<asp:TextBox ID="Txtcpasswoed" runat="server"></asp:TextBox>
<asp:CompareValidator ID="CompareValidator1" runat="server"
    ControlToCompare="Txtpassword" ControlToValidate="Txtcpasswoed"
    ErrorMessage="confirm password not same as passord"
    ToolTip="not same as password" Type="Integer"></asp:CompareValidator>
<br />
<br />
sem<asp:TextBox ID="Txtsem" runat="server"></asp:TextBox>
<asp:RangeValidator ID="RangeValidator1" runat="server"
    ControlToValidate="Txtsem" ErrorMessage="not valid semester"
    MaximumValue="8"
    MinimumValue="1"></asp:RangeValidator>
<br />
<asp:Button ID="Button1" runat="server" onclick="Button1_Click" Text="submit"/>
<br />
<asp:ValidationSummary ID="ValidationSummary1" runat="server" />
</form>
</body>
</html>
```

## OUTPUT:

name

email

phone no  \*

password

confirm password

sem

- not valid phone no

## Practical 8

AIM:

### Introduction to Master Pages.

#### Site1.Master

```
<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="Site1.master.cs"
Inherits="ASPApplication2.Site1" %>
```

```
<!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">
        <table>
            <tr> <td>
                <asp:Label ID="lblheader" runat="server"
Text="header"></asp:Label></td></tr>
            <tr>
                <td>
                    <asp:Button ID="Buttonsearch" runat="server" Text="Button" />
                    <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
                    </asp:ContentPlaceHolder>
                </td>
            </tr>
            <tr><td>footer</td></tr>
        </table>
    </form>
</body>
</html>
```

#### Site1.Master.cs

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

namespace ASPApplication2
{
    public partial class Site1 : System.Web.UI.MasterPage
    {
    }
```

```

        protected void Page_Load(object sender, EventArgs e)
        {

        }

        public Label lblHeader
        {
            get { return lblheader; }
        }

        public Button buttonsearch
        {
            get { return Buttonsearch; }
        }
    }
}

```

### WebForm1.aspx

```

<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master"
AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="ASPApplication2.WebForm1" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
runat="server">
    <asp:TextBox ID="txtHeader" runat="server"></asp:TextBox>
    <asp:Button ID="btn1" runat="server" Text="button"
        onclick="Button1_Click" />
</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 ASPApplication2
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            ((Site1)Master).LblHeader.Text = txtHeader.Text;
        }
    }
}

```

OUTPUT:

hello



footer

### WebForm2.aspx

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master"
AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs"
Inherits="ASPApplication2.WebForm2" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1"
runat="server">
    <asp:GridView ID="getdetails" 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 ASPApplication2
{
    public partial class WebForm2 : System.Web.UI.Page
    {
        protected void Page_Init(object sender, EventArgs e)
        {
            ((Site1)Master).buttonsearch.Click+=new
            EventHandler(buttonsearch_Click);
        }
        void buttonsearch_Click(object sender, EventArgs e)
        {
            getData();
        }
        protected void Page_Load(object sender, EventArgs e)
        {
        }
        void getData()
        {
            string source = @"Data Source=CE3COMP3\squlexpress;Initial
            Catalog=DBstudent;Integrated Security=True;Pooling=False";
            string select = "select * from Tbl1";
            SqlConnection conn = new SqlConnection(source);
            SqlCommand cmd = new SqlCommand(select,conn);
```

```
        conn.Open();
        SqlDataReader reader = cmd.ExecuteReader();
        getdetails.DataSource = reader;
        getdetails.DataBind();
        conn.Close();
    }
}
```

OUTPUT:

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