



# **LAB Manual** **.Net Technology**

Samir Katyar

170473107010

VVP Engineering College,Rajkot

## Contents

AIM : Introduction to C# .....	1
Program 1.....	1
AIM: Inheritance.....	10
Program 1.....	10
Program 2.....	11
Program 3.....	13
Program 4.....	14
AIM: Method & constructor overloading .....	17
Program 1.....	17
Program 2.....	22
AIM: Reflection .....	25
Program 1.....	25
AIM: Files Operations .....	29
Program 1.....	29
Program 2.....	31
Program 3.....	33
AIM: Student Registration.....	37
Program 1.....	37
AIM: Validation Controls .....	40
Program 1.....	40
AIM: Master Page.....	43
Program 1.....	43

## Practical 1

### AIM : Introduction to C#

#### Variables:

- Initialization

- Scope

- Constant

#### Predefined Data Types

- Value Types

- Reference Types

#### Flow Control

- Conditional Statements(if, switch)

- Loop(for, while, dowhile, foreach)

- Jump(goto, break, continue, return)

#### Eumerations

#### Passing Arguments

### Program 1

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

namespace aim
{
    class Program
```

```
{  
  
    static int newint=100;  
  
    public enum TimeOfDay  
    {  
        Morning = 0,  
        Afternoon = 1,  
        Evening = 2  
    }  
  
    public static void Main(string[] args)  
    {  
        Console.WriteLine("\n integer types");  
  
        sbyte sb = 10;  
        short s = 33;  
        int i = 10;  
        long l = 33L;  
        byte b = 22;  
        ushort us = 33;  
        uint ul = 33u;  
        ulong ulo = 33ul;  
  
        Console.WriteLine("{0},{1},{2},{3},{4},{5},{6},{7}", sb, s, i, l, b, us,  
ul, ulo);  
  
        float f = 1.122345656767f;  
        double d = 12.1234455657878797;  
  
        Console.Write("\nFloat and Double:\n");  
  
        Console.WriteLine("{0} and \n{1}", f, d);  
  
        decimal dec=111.6666666666666666666666M;  
  
        Console.WriteLine("decimal:\n{0} ",dec);  
  
        Console.WriteLine("\nBoolean:");  
    }  
}
```

```
bool boolean =true;

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

// Console.ReadLine();

char character ='d';

Console.WriteLine(character);

character = '\0';

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

object o1 = "Hi, I am ALICE";

object o2 = 15.3454365;

string strObj = o1 as string;

Console.WriteLine(strObj);

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

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

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

string s1, s2;

s1 = "this is string";

s2 = s1;

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

s2 = "other string";

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

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

Console.WriteLine(s1);

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

Console.WriteLine(s1);

s1 = @"We can also write

like this";

Console.WriteLine(s1);

bool isZero;
```

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

if (i == 10)
{
    isZero = true;
    Console.WriteLine("i is Zero {0}",isZero);
}
else
{
    isZero = false;
    Console.WriteLine("i is Non - zero");
}

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

```
WriteGreeting(TimeOfDay.Morning);

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


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

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

int newint=0;

int j;

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

```

```
//int j;

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

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

//used in a 'parent or current' scope to denote something else"
Console.WriteLine("{0} {1}\n", newint, Program.newint);
}

    Console.WriteLine("2:");
for (int k = 0; k < 3; k++)
{
    Console.WriteLine("{0} ", k);
} //Scope of k ends here
Console.WriteLine("\n");
//Console.WriteLine(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.WriteLine("{0} ", k);
} //scope of k ends here again

Console.WriteLine("Constants");

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

//valConst = 45;

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



```
//const only allow constant variables into the expression

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

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

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


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

//Value Types

int vali = 2, valj = vali;

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

valj = 90;

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

//Referece Types

Vector x, y;

x = new Vector();

x.value = 3;


y = x;


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

y.value = 234;

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

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

y = null;

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

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

```
        //CTS

    }

    public class Vector
    {
        public int value;
    }
}

}
```

**OutPut:**

```
C:\Users\Dell\source\repos\firstprogram\firstprogram\bin\Debug\firstprogram....
```

```
integer types  
10,33,10,33,22,33,33,33  
  
Float and Double:  
1.122346 and  
12.1234455657879  
decimal:  
111.666666666666666666666666666666  
  
Boolean:  
Status: True  
d  
Now null:  
Hi, I am ALICE  
1487365606 System.String  
1302462624 System.Double  
False  
S1 is: this is string and s2 is this is string  
S1 is: this is string and s2 is other string  
c:C:\Users\Dell\source\repos\aim  
c:C:\Users\Dell\source\repos\aim\aim  
We can also write  
like this  
  
Flow Control: <if>  
i is 10  
i is Zero True  
  
Switch:  
integerA = 1  
Good morning!  
Scope of Variables.  
1:  
0 100  
0 100  
2:  
0 1 2  
3 2 1 Constants  
100 is constant value  
Another Constant: 109  
  
Predefined Data Types  
  
Value Types and Reference Types  
val i is: 2 and val j is: 2  
val i is: 2 and val j is: 90  
x is: 3 and y is: 3  
x is: 234 and y is: 234
```

## Practical 2

### AIM: Inheritance

#### Program 1

Perform following programs in c#.

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

@ @ @ @ @

@ @ @ @

@ @ @

@ @

@

```
using System;
```

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

```
using System.Linq;
```

```
using System.Text;
```

```
using System.Threading.Tasks;
```

```
namespace practical2
```

```
{
```

```
    class Program
```

```
    {
```

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

```
        {
```

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

```
            {
```

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

```
                {
```

```
                    Console.Write("@");
```

```

        }

        Console.WriteLine(" ");
    }

    Console.ReadKey();
}
}
}

```

### **OutPut:**

```

C:\dotNet\Practical2(1)\ConsoleApp5>csc Pattern1.cs
Microsoft (R) Visual C# Compiler version 2.10.0.0 (b9fb1610)
Copyright (C) Microsoft Corporation. All rights reserved.

C:\dotNet\Practical2(1)\ConsoleApp5>Pattern1.exe
@@@@
@@@
@@
@
-

```

## Program 2

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

```

1
1 2
1 2 3
1 2 3 4

```

```

using System;

using System.Collections.Generic;

```

```
using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace practical2._1
{
    class Program
    {
        static void Main(string[] args)
        {
            for(int i=1;i<5;i++)
            {
                for(int j=1;j<=i;j++)
                {
                    Console.Write(j+" ");

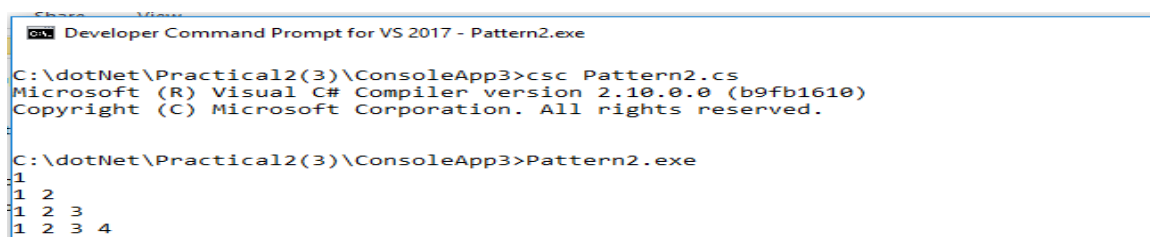
                }

                Console.WriteLine();

            }

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

OutPut:



```
Developer Command Prompt for VS 2017 - Pattern2.exe
C:\dotNet\Practical2(3)\ConsoleApp3>csc Pattern2.cs
Microsoft (R) Visual C# Compiler version 2.10.0.0 (b9fb1610)
Copyright (C) Microsoft Corporation. All rights reserved.

C:\dotNet\Practical2(3)\ConsoleApp3>Pattern2.exe
1
1 2
1 2 3
1 2 3 4
```

### Program 3

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

Hello Ram from country India

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace practical2._2
{
    class Program
    {
        static void Main(string[] args)
        {
            string name;

            string country;

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

            name=Console.ReadLine();

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

            country = Console.ReadLine();

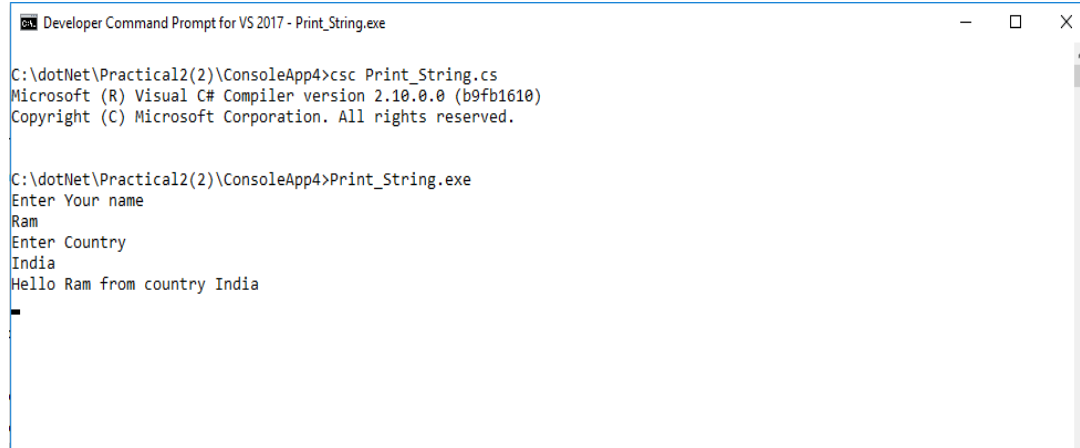
            Console.WriteLine("hello {0} from country {1}",name,country);

            Console.ReadKey();

        }

    }
```

```
}
```

**OutPut:**

```
Developer Command Prompt for VS 2017 - Print_String.exe

C:\dotNet\Practical2(2)\ConsoleApp4>csc Print_String.cs
Microsoft (R) Visual C# Compiler version 2.10.0.0 (b9fb1610)
Copyright (C) Microsoft Corporation. All rights reserved.

C:\dotNet\Practical2(2)\ConsoleApp4>Print_String.exe
Enter Your name
Ram
Enter Country
India
Hello Ram from country India
```

**Program 4**

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

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

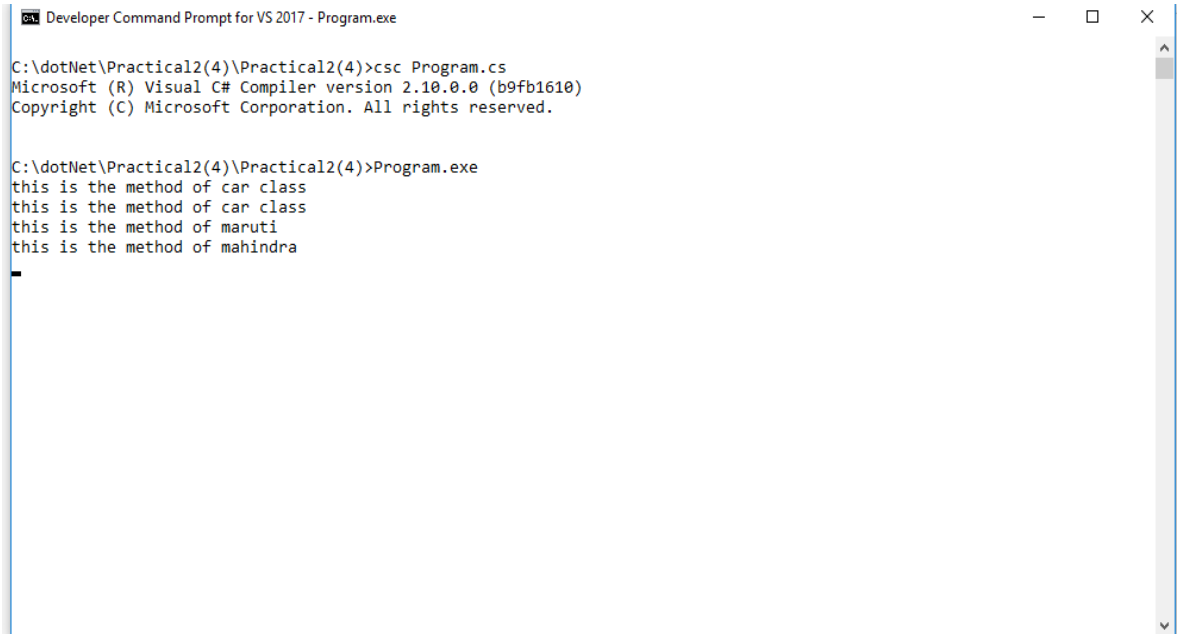
namespace practical2._3
{
    class car
    {
        public void Method1()
        {
            Console.WriteLine("this is the method of car class");
        }
    }
}
```



```
    }  
}  
class maruti:car  
{  
    public void method2()  
    {  
        Console.WriteLine("this is the method of maruti");  
        Console.ReadKey();  
    }  
}  
class mahindra:car  
{  
    public void method3()  
    {  
        Console.WriteLine("this is the method of mahindra");  
    }  
}  
class Program  
{  
    static void Main(string[] args)  
    {  
        mahindra m = new mahindra();  
        maruti m1 = new maruti();  
        m.Method1();  
        m1.Method1();  
        Console.ReadKey();  
    }  
}
```

```
}
```

OutPut:



Developer Command Prompt for VS 2017 - Program.exe

```
C:\dotNet\Practical2(4)\Practical2(4)>csc Program.cs
Microsoft (R) Visual C# Compiler version 2.10.0.0 (b9fb1610)
Copyright (C) Microsoft Corporation. All rights reserved.

C:\dotNet\Practical2(4)\Practical2(4)>Program.exe
this is the method of car class
this is the method of car class
this is the method of maruti
this is the method of mahindra
```

## Practical 3

### AIM: Method & constructor overloading

#### Program 1

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

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Practical3

{

    class Program

    {

        public void add(int a, int b)

        {

            int sum = a + b;

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

        }

        public void add()

        {

            int i, j, n;

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

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

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

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

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

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

for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.Write("{0},{1}:", i, j);

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

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

for (i = 0; i < n; i++)
{
    for (j = 0; j < n; j++)
    {
        Console.Write("{0},{1}:", i, j);

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

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

for (i = 0; i < n; i++)
{
    Console.Write("\n");

    for (j = 0; j < n; j++)
        Console.Write("{0}\t", arr1[i, j]);
}

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

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

```
{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)
        Console.WriteLine("{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.WriteLine("\nAddition of Two Matrix:\n");
for (i = 0; i < n; i++)
{
    Console.WriteLine("\n");
    for (j = 0; j < n; j++)
    {
        Console.WriteLine("{0}\t", crr1[i, j]);
    }
}
}

public void add(Vector a, Vector b)
{
    Vector result=new Vector();
    result.x = a.x + b.x;
    result.y = a.y + b.y;
    result.z = a.z + b.z;
```

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

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

    }

    static void Main(string[] args)
    {
        Program p = new Program();

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

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

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

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

        p.add(a, b);

        p.add();

        Vector v1 = new Vector();

        Vector v2 = new Vector();

        // float x, y, z;

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

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

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

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

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

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

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

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

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

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

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

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

```

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

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

        p.add(v1, v2);

        Console.ReadLine();

    }

}

public class Vector

{

    public float x, y,z;

}

}

```

### **OutPut:**

```

Developer Command Prompt for VS 2017 - Overloading.exe
Value of b:
1
Addition is:3
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 :
0,0:5
0,1:6
1,0:7
1,1:8

The First matrix is :

1      2
3      4
The Second matrix is :

5      6
7      8
Addition of Two Matrix:

6      8
10     12
Enter 1st vector
X:
1
Y:
2
Z:
3

Enter 2nd vector
X:
4
Y:
5
Z:
6
Addition of Two vectors is:
<5,7,9>

```

## Program 2

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

1. Name

2. Name, Enrollment

3. Name, Enrollment, Branch

```
using System;
```

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

```
using System.Linq;
```

```
using System.Text;
```

```
using System.Threading.Tasks;
```

```
namespace P3_2_
```

```
{
```

```
    class Program
```

```
    {
```

```
        public int ID { get; set; }
```

```
        public string Name { get; set; }
```

```
        String name, branch;
```

```
        int enroll;
```

```
        Program(String Stname)
```

```
        {
```

```
            name = Stname;
```

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

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

```
        }
```

```
        Program(String Stname,String Stbranch)
```



```
{
    name = Sname;
    branch = Stbranch;
    Console.WriteLine("2nd Constructor:");
    Console.WriteLine(Sname+" is in "+Stbranch+" branch");
}
Program(String Sname, String Stbranch ,int Stenroll)
{
    name = Sname;
    branch = Stbranch;
    enroll = Stenroll;
    Console.WriteLine("3rd Constructor:");
    Console.WriteLine(Sname + " is in " + Stbranch+" having "+Stenroll+"
Enrollment ");
}
static void Main(string[] args)
{
    Program p = new Program("bob");
    Program p1 = new Program("bob",1);
    Program p2 = new Program("bob",1,"Computer");
    Console.ReadLine();

}
}
}
```

**OutPut:**

## Practical 4

### AIM: Reflection

#### Program 1

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

namespace p3a1
{
    class Program
    {
        public int ID { get; set; }
        public string Name { get; set; }
        String name, branch;
        int enrol;

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

        public void printName()
        {
            Console.WriteLine("Name is: {0}", this.Name);
        }
    }
}
```

```
    }

    public Program(String name)
    {
        this.name = name;

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

    public Program(String name, int enrol)
    {
        this.name = name;

        this.enrol = enrol;

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

    public Program(String name, int enrol, String branch)
    {
        this.name = name;

        this.enrol = enrol;

        this.branch = branch;

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

    static void Main(string[] args)
    {
try
        {
            Type T = Type.GetType("p3a1.Program");

            MethodInfo[] methods = T.GetMethods();

            foreach (MethodInfo method in methods)
```

```
{
    Console.WriteLine(method.ReturnType + " " + method.Name);
}

PropertyInfo[] properties = T.GetProperties();

Console.WriteLine("\nProperties");
foreach (PropertyInfo property in properties)
{
    Console.WriteLine(property.PropertyType + " " + property.Name);
}

Console.WriteLine("\nConstructors");
ConstructorInfo[] constructors = T.GetConstructors();
foreach (ConstructorInfo constructor in constructors)
{
    Console.WriteLine(constructor.ToString());
}

Program p1 = new Program("bob");
Program p2 = new Program("bob", 1);
Program p3 = new Program("bob", 1, "computer");

Console.ReadLine();

catch (Exception e)
{

```

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

## Practical 5

### AIM: Files Operations

#### Program 1

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

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Practical5
{
    class Program
    {
        static void Main(string[] args)
        {
            CopyFile cp = new CopyFile();
            String file1= @"C:\dotNet\file1.txt";
            String file2 = @"C:\dotNet\richa\file2.txt";
            cp.copyFile(file1, file2);
        }
    }

    public class CopyFile
    {
        public void copyFile(String file1,String file2)
```

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

**OutPut:**



## Program 2

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

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Practical5_1_
{
    class Program
    {
        static void Main()
        {
            StreamReader reader = new StreamReader("teststream.txt");

            using (reader)
            {
                int lineNumber = 0;

                String line = reader.ReadLine();

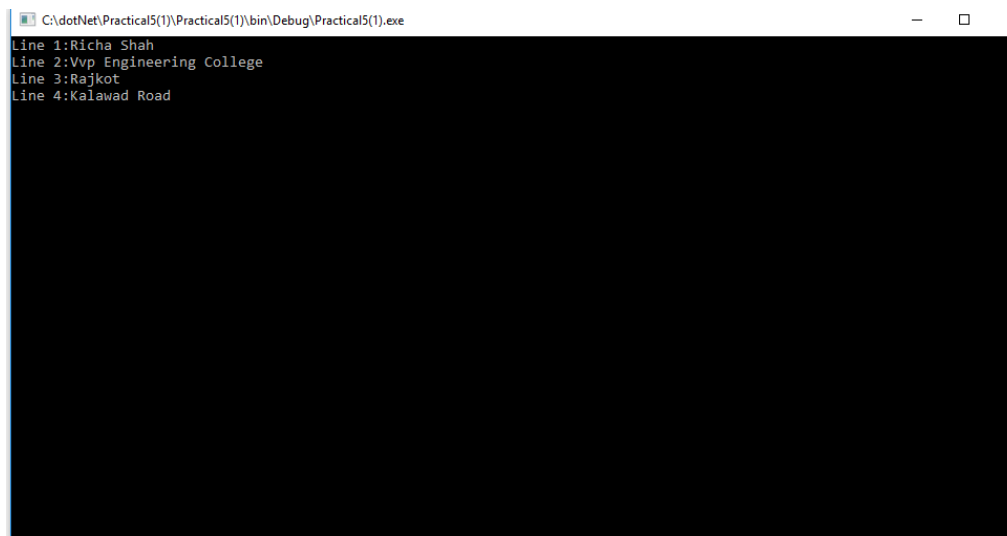
                while(line!=null)
                {
                    lineNumber++;

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

                    line = reader.ReadLine();
                }

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

```
        }  
    }  
}
```

**Output:**

```
C:\dotNet\Practical5(1)\Practical5(1)\bin\Debug\Practical5(1).exe  
Line 1:Richa Shah  
Line 2:Vvp Engineering College  
Line 3:Rajkot  
Line 4:Kalawad Road
```

### Program 3

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

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace Practical5_2_
{
    class Program
    {
        static void Main(string[] args)
        {
            string[] Directories =
Directory.GetDirectories(@"C:\Users\RICHA\source\repos");

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

            foreach (string dir in Directories)
            {
                //Console.WriteLine("All the Directories are:");

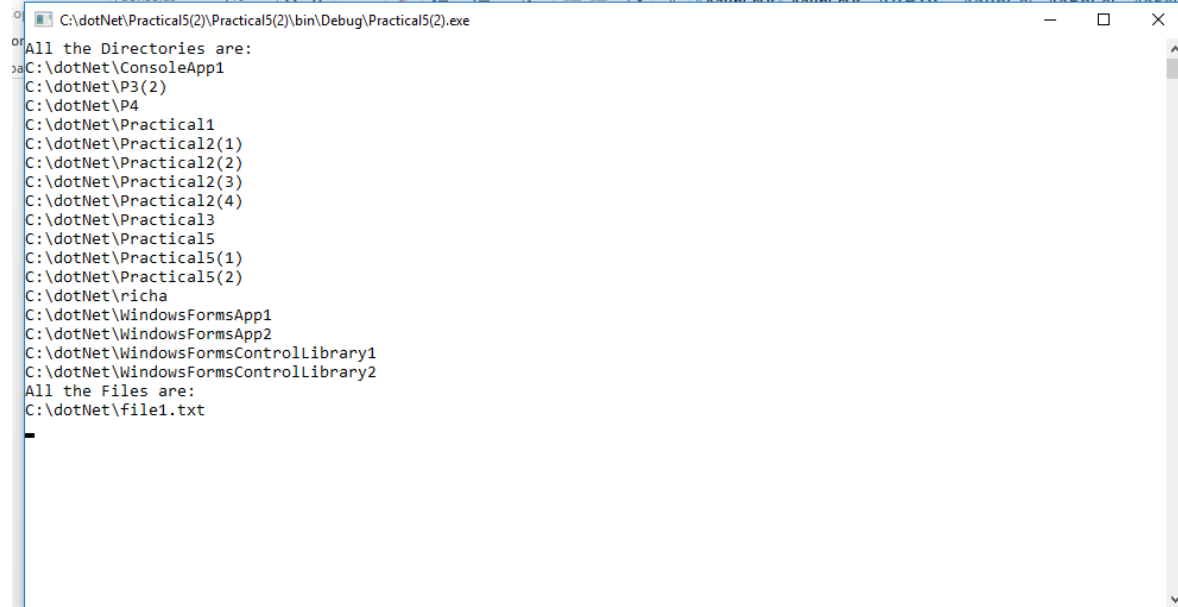
                Console.WriteLine(dir);
            }

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

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

            foreach (string file in files)
            {
                // Console.WriteLine("All the Files are:");
            }
        }
    }
}
```

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

**Output:**

A screenshot of a Windows command prompt window. The title bar shows the file path: C:\dotNet\Practical5(2)\Practical5(2)\bin\Debug\Practical5(2).exe. The window contains the following text:

```
Or All the Directories are:  
C:\dotNet\ConsoleApp1  
C:\dotNet\P3(2)  
C:\dotNet\P4  
C:\dotNet\Practical1  
C:\dotNet\Practical2(1)  
C:\dotNet\Practical2(2)  
C:\dotNet\Practical2(3)  
C:\dotNet\Practical2(4)  
C:\dotNet\Practical3  
C:\dotNet\Practical5  
C:\dotNet\Practical5(1)  
C:\dotNet\Practical5(2)  
C:\dotNet\richa  
C:\dotNet\WindowsFormsApp1  
C:\dotNet\WindowsFormsApp2  
C:\dotNet\WindowsFormsControllLibrary1  
C:\dotNet\WindowsFormsControllLibrary2  
All the Files are:  
C:\dotNet\file1.txt
```



## Practical 6

### AIM: Student Registration

#### Program 1

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

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.IO;

namespace P6_form_
{
    public partial class Form1 : Form
    {
        string imgPath;
        public Form1()
        {
            InitializeComponent();
        }

        private void label1_Click(object sender, EventArgs e)
        {
        }

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

        private void button3_Click(object sender, EventArgs e)
        {
            Environment.Exit(0);
        }

        private void button2_Click(object sender, EventArgs e)
        {
            string source = @"C:\DOTNET\P6(FORM)\P6(FORM)\PROPERTIES\DATABASE1.MDF";
            string select = "select count(*) from tblStudent";
            SqlConnection conn = new SqlConnection(source);
            SqlCommand cmd = new SqlCommand(select, conn);
            conn.Open();
            int i = Convert.ToInt16(cmd.ExecuteScalar());
            int textBox1 = i + 1;
        }
    }
}
```

```

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

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

private void button1_Click(object sender, EventArgs e)
{
    openFileDialog1.Filter = "Jpg|*.jpg";
    if (openFileDialog1.ShowDialog() == DialogResult.OK)
    {
        imgPath = @"C:\Pictures" + openFileDialog1.SafeFileName;
        imgStudent.Image = Image.FromFile(openFileDialog1.FileName);
    }
}
}
}

```

### Output:

First Name: ABC

Last Name: AAA

Gender: ☐ Male ☒ Female

subject: ☒ s1 ☐ s2

Upload

Save





## Practical 7

### AIM: Validation Controls

#### Program 1

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

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

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

```
<head runat="server">
```

```
<title></title>
```

```
</head>
```

```
<body style="height: 19px">
```

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

```
<p>
```

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

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

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

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

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

```
</p>
```

```
<p>
```

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

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

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

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

```

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

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

</p>

<p>

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

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

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

            ControlToCompare="txtPass" ControlToValidate="txtConfirm"

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

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

</p>

<p>

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

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

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

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

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

</p>

<p>

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

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

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

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

</p>

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

```

```
<asp:ValidationSummary ID="ValidationSummary1" runat="server" />  
</form>  
</body>  
</html>
```

**Output:**

Name	<input type="text"/>	RequiredFieldValidator
Email	<input type="text" value="abcde"/>	RegularExpressionValidator
Password	<input type="password" value="..."/>	
Confirm Password	<input type="password" value="..."/>	CompareValidator
Sem	<input type="text" value="9"/>	RangeValidator

- RequiredFieldValidator
- RegularExpressionValidator
- CompareValidator
- RangeValidator

## Practical 8

### AIM: Master Page

#### Program 1

##### **Webform2.cs:**

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Data.SqlClient;

namespace WebApplication5

{

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

    {

        protected void Page_Init(object sender, EventArgs e)

        {

            ((Site1)Master).BtnSearch.Click += new EventHandler(btnSearch_Click);

        }

        protected void btnSearch_Click(object sender, EventArgs e)

        {

            GetData();

        }

        protected void Page_Load(object sender, EventArgs e)
```

```
{  
  
}  
  
void GetData()  
  
{  
  
    string source = @"Data  
Source=.\SQLEXPRESS;AttachDbFilename=C:\Users\cecomp1\Documents\emp.mdf;Integrated  
Security=True;Connect Timeout=30;User Instance=True";  
  
    string select ="select * from tblStudent";  
  
    SqlConnection conn = new SqlConnection(source);  
  
    SqlCommand cmd = new SqlCommand(select, conn);  
  
    conn.Open();  
  
    SqlDataReader reader = cmd.ExecuteReader();  
  
    grdEmp.DataSource = reader;  
  
    grdEmp.DataBind();  
  
    conn.Close();  
  
}  
  
}  
}
```

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

**Webform1.cs**

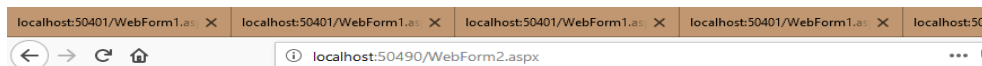
```

namespace WebApplication5
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

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

```

**Output:**

Header

Name	RollNo.	Semester
Richa	16ce026	6
Divya	16ce037	6
Bhargav	16ce012	6
Jarna	16ce055	5
Akash	16ce043	4
Search		

Footer

## Practical 9

### AIM: Web Services

#### Program 1

### Create web service & consume it

#### WebService1.asmx.cs:

```
using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Services;

namespace Service {

[WebService(Namespace = "http://tempuri.org/")] [WebServiceBinding(ConformsTo =
WsiProfiles.BasicProfile1_1)] [System.ComponentModel.ToolboxItem(false)]

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;
```

```
}
```

```
}
```

```
}
```

## WebForm1.aspx:

```
<%@ Page Language="C#" AutoEventWireup="true"
```

```
CodeBehind="WebForm1.aspx.cs" Inherits="WebService.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>

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

</asp:RequiredFieldValidator>

    <asp:RegularExpressionValidator ID="RegularExpressionValidator2" runat="server"
ControlToValidate="txtA" ErrorMessage="RegularExpressionValidator"
ValidationExpression="^[0-9]+">

</asp:RegularExpressionValidator>

<br />

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

</asp:RequiredFieldValidator> <asp:RegularExpressionValidator
ID="RegularExpressionValidator1" runat="server" ControlToValidate="txtB"
ErrorMessage="RegularExpressionValidator" ValidationExpression="^[0-
9]+"></asp:RegularExpressionValidator>

<br />

<asp:Button ID="btnadd" runat="server" onclick="btnadd_Click" Text="Add" />

    <asp:Button ID="btnsub" runat="server" onclick="btnsub_Click" Text="Sub" />

    <asp:Button ID="btnmul" runat="server" onclick="btnmul_Click" Text="Mul" />

<asp:Button ID="btndiv" runat="server" onclick="btndiv_Click" Text="Div" /> <br />

    <asp:Label ID="lblresult" runat="server" Text="Result">

</asp:Label>

</div>

</form>
```

</body>

</html>

## 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 WebService {

    public partial class WebForm1 : System.Web.UI.Page { localhost.WebService1 calc = new
localhost.WebService1();

        protected void Page_Load(object sender, EventArgs e) {

        }

        protected void btnadd_Click(object sender, EventArgs e)

        {

            lblresult.Text = calc.Add(Convert.ToInt16(txtA.Text),
            Convert.ToInt16(txtB.Text)).ToString();

        }

        protected void btnsub_Click(object sender, EventArgs e)

        {

            lblresult.Text = calc.Sub(Convert.ToInt16(txtA.Text),
            Convert.ToInt16(txtB.Text)).ToString();

        }

        protected void btnmul_Click(object sender, EventArgs e) {
```

```
lblresult.Text = calc.Mul(Convert.ToInt16(txtA.Text),  
Convert.ToInt16(txtB.Text)).ToString();
```

```
}
```

```
protected void btndiv_Click(object sender, EventArgs e) {
```

```
lblresult.Text = calc.Div(Convert.ToInt16(txtA.Text),  
Convert.ToInt16(txtB.Text)).ToString();
```

```
}
```

```
}
```

```
}
```

### Output:



1  
3  
Add Sub Mul Div  
4