Table of Contents

[Lab 1: Implementation of Class and Methods 1](#_Toc185424908)

[Lab 2: Implementation of Array 3](#_Toc185424909)

[Lab 3: Implementation of If and Switch Statements 5](#_Toc185424910)

[Lab 4: Implementation of For and Foreach Loop 7](#_Toc185424911)

[Lab 5: Implementation of While and Do-While Loop 8](#_Toc185424912)

[Lab 6: Implementation of Constructors 10](#_Toc185424913)

[Lab 7: Implementation of Inheritance 11](#_Toc185424914)

[Lab 8: Implementation of Abstract Class and Method 12](#_Toc185424915)

[Lab 9: Implementation of Method Overloading 13](#_Toc185424916)

[Lab 10: Implementation of Interface 14](#_Toc185424917)

[Lab 11: Implementation of Generics 15](#_Toc185424918)

[Lab 12: Implementation of Delegates 16](#_Toc185424919)

[Lab 13: Implementation of Lambda Expression 17](#_Toc185424920)

[Lab 14: Implementation of Exception Handling 18](#_Toc185424921)

[Lab 15: Implementation of LINQ 19](#_Toc185424922)

[Lab 16: Implementation of Web Forms 20](#_Toc185424923)

[Lab 17: Implementation of Database 22](#_Toc185424924)

# Lab 1: Implementation of Class and Methods

**Objective:**

* To implement class and methods

**Source Code:**

using System;

namespace Project1

{

    internal class Class1

    {

        public class Calculation

        {

            //non parameterized method

            public void Add()

            {

                Console.WriteLine("This is non-parameterized method");

            }

            //parameterized method

            public void Add(int x, int y)

            {

                Console.WriteLine("This is parameterized method.\nSum:"+(x+y));

            }

        }

        static void Main(string[] args)

        {

            Calculation cal = new Calculation();

            cal.Add();

            cal.Add(10,20);

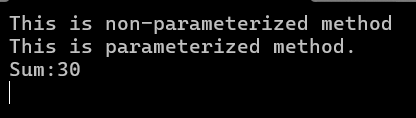
            Console.ReadLine();

        }

    }

}

**Output:**

****

**Conclusion:**

From this lab, we learned to implement class and methods. Hence, a class Calculation is created and parameterized and non-parameterized methods are created and output is displayed.

# Lab 2: Implementation of Array

**Objective:**

* To implement array

**Source Code:**

using System;

namespace Chapter\_Two\_Arrays

{

    internal class Program

    {

        static void Main(string[] args)

        {

            //Infinite length Array

            string[] cars;

            cars = new string[] { "Tiago", "Nexon", "Polo", "Volkswagen" };

            //Fixed length Array

            string[] schools = new string[5];

            schools[0] = "Divya Gyan";

            schools[1] = "Gyankunj";

            schools[2] = "Islington";

            schools[3] = "St. Xavier";

            schools[4] = "St. Marys";

            Console.WriteLine("---Infinite Length Array---");

            foreach (string s in cars) {

                Console.WriteLine(s);

            }

            Console.WriteLine("---Fixed Length Array---");

            foreach (string s in schools) {

                Console.WriteLine(s);

            }

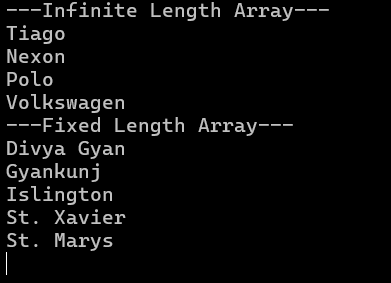
            Console.ReadLine();

        }

    }

}

**Output**:



**Conclusion:**

From this lab, we learned to implement class and methods. Hence, a class Calculation is created and parameterized and non-parameterized methods are created and output is displayed.

# Lab 3: Implementation of If and Switch Statements

**Objective:**

* To implement If and Switch statements

**Source Code of If Statements:**

using System;

namespace Chapter\_Two\_If\_Statement

{

    internal class Program

    {

        static void Main(string[] args)

        {

            int number = 15;

            if (number > 20)

            {

             Console.WriteLine("The number is greater than 20.");

            }

else if (number > 10)

            {

            Console.WriteLine("The number is greater than 10 but less than or equal to 20.");

            }

            else

            {

                Console.WriteLine("The number is 10 or less.");

            }

            Console.ReadLine();

        }

    }

}

**Output:**

****

**Source Code of Switch Statements:**

using System;

namespace Chapter\_Two\_If\_Statement

{

    internal class Program

    {

        static void Main(string[] args)

        {

            int iNo;

            Console.WriteLine("Enter a Number: ");

            iNo = Convert.ToInt32(Console.ReadLine());

            switch (iNo) {

                case 100:

                    Console.WriteLine("Value is 100");

                    break;

                case 50:

                    Console.WriteLine("Value is 50");

                    break;

                case 25:

                    Console.WriteLine("Value is 25");

                    break;

                default:

                    Console.WriteLine("Value is "+iNo);

                    break;

            }

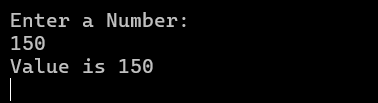
            Console.ReadLine();

        }

    }

}

**Output:**

****

## Conclusion:

From this lab, we learned to implement if-else and switch cases. Hence, a program was written using if-else and another program was written using the switch case.

# Lab 4: Implementation of For and Foreach Loop

**Objective:**

* To implement for and foreach loops

**Source Code:**

using System;

namespace Chapter\_Two\_For\_Each\_Loop

{

    internal class Program

    {

        static void Main(string[] args)

        {

            string[] strMotors = { "Nexon", "BYD", "Thar", "Hilux" };

            Console.WriteLine("Using ForEach Loop");

            foreach (string s in strMotors)

                Console.WriteLine(s);

            Console.WriteLine("\nUsing For Loop");

            for (int i = 0; i < strMotors.Length; i++)

                Console.WriteLine(strMotors[i]);

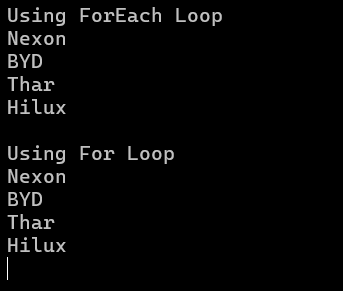
            Console.ReadLine();

        }

    }

}

**Output**:



**Conclusion:**

From this lab, we learned to implement for and foreach loop. Hence, elements of an array are displayed using for and foreach loop.

# 

# Lab 5: Implementation of While and Do-While Loop

**Objective:**

* To implement for and foreach loops

**Source Code:**

using System;

namespace Chapter\_Two\_Do\_While\_Loop

{

    internal class Program

    {

        static void Main(string[] args)

        {

            int iNo,iNo1;

            Console.WriteLine("Enter a number: ");

            iNo=Convert.ToInt32(Console.ReadLine());

            iNo1 = iNo;

            Console.WriteLine("---Using While Loop---");

            while (iNo > 0)

            {

                Console.WriteLine(iNo);

                iNo--;

            }

            Console.WriteLine("\n---Using Do-While Loop---");

            do

            {

                Console.WriteLine(iNo1);

                iNo1--;

            }while(iNo1 > 0);

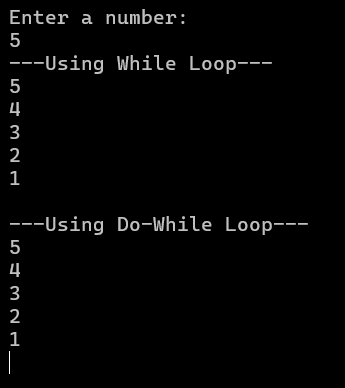
            Console.ReadLine();

        }

    }

}

**Output:**

****

**Conclusion:**

From this lab, we learned to implement while and do-while loop. Hence, the numbers are displayed using while and do-while loop.

# Lab 6: Implementation of Constructors

**Objective:**

* To implement constructors

**Source Code:**

using System;

namespace Chapter\_Three\_Constructors

{

    internal class Program

    {

        public static int iRollNo;

        public static string myName;

        public int MyClass;

        static Program()

        {

            iRollNo = 2;

            myName = "Hari";

        }

        public Program()

        {

            MyClass = 10;

        }

        static void Main(string[] args)

        {

            Program p = new Program();

            Console.WriteLine(iRollNo);

            Console.WriteLine("My name is " + myName);

            Console.WriteLine("I study in class " + p.MyClass);

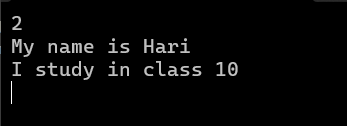
            Console.ReadLine();

        }

    }

}

**Output:**



**Conclusion:**

From this lab, we learned to implement static and non-static constructors. Hence, the constructors are created and the values are displayed.

# Lab 7: Implementation of Inheritance

**Objective:**

* To implement inheritance

**Source Code:**

using System;

namespace Chapter\_Three\_Inheritance

{

    internal class Program

    {

        public class Animal

        {

            public void Sound(){

                Console.WriteLine("Animal makes sound.");

            }

        }

        public class Dog: Animal

        {

            public void MakeSound(){

                Console.WriteLine("Dog Barks.");

            }

        }

        static void Main(string[] args)

        {

            Dog dog = new Dog();

            dog.Sound();

            dog.MakeSound();

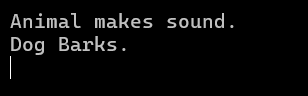
            Console.ReadLine();

        }

    }

}

**Output:**

****

## Conclusion:

From this lab, we learned to implement Inheritance. Hence, the program was created using implementing the Inheritance.

# Lab 8: Implementation of Abstract Class and Method

**Objective:**

* To implement abstract class and methods

**Source Code:**

using System;

namespace Chapter\_Three\_Abstract\_Class

{

    internal class Program

    {

        public abstract class MyClass

        {

            public abstract void Enroll();

        }

        public class School : MyClass

        {

            public override void Enroll()

            {

                Console.WriteLine("Ram Enrolled in 5th Semester");

            }

        }

        static void Main(string[] args)

        {

            School s = new School();

            s.Enroll();

            Console.ReadLine();

        }

    }

}

**Output:**



**Conclusion:**

From this lab, we learned to implement abstract class and method. Hence, an abstract class and method are created and displayed.

# Lab 9: Implementation of Method Overloading

**Objective:**

* To implement method overloading

**Source Code:**

using System;

namespace Chapter\_Three\_Abstract\_Class

{

    internal class Program

    {

        public class Calculation

        {

            public void Add(int x, int y)

            {

                Console.WriteLine("Sum: " + (x + y));

            }

            public void Add(int x, int y,int z)

            {

                Console.WriteLine("Sum: " + (x + y + z));

            }

        }

        static void Main(string[] args)

        {

            Calculation calc = new Calculation();

            calc.Add(1, 2);

            calc.Add(2, 3, 4);

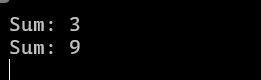
            Console.ReadLine();

        }

    }

}

**Output:**



**Conclusion:**

From this lad, we learned to implement method overloading. Hence, two methods Add with different parameters are created and the results are displayed.

# Lab 10: Implementation of Interface

**Objective:**

* To implement interface

**Source Code:**

using System;

namespace Chapter\_Three\_Interface

{

    internal class Program

    {

        interface IAnimal

        {

            void MakeSound();

        }

        class Dog : IAnimal

        {

            public void MakeSound()

            {

                Console.WriteLine("Dog says: Woof!");

            }

        }

        static void Main(string[] args)

        {

            IAnimal dog = new Dog();

            dog.MakeSound();

            Console.ReadLine();

        }

    }

}

**Output:**



**Conclusion:**

From this lab, we learned about interface. Hence, an interface Animal is created and it is implemented in class Dog and displayed.

# Lab 11: Implementation of Generics

**Objective:**

* To implement generics

**Source Code:**

using System;

namespace Chapter\_Three\_Generic

{

    internal class Program

    {

        class Box<T>

        {

            public T Value { get; set; }

        }

        static void Main(string[] args)

        {

            Box<int> intBox = new Box<int> { Value = 42 };

            Box<string> stringBox = new Box<string> { Value = "Hello, Generics!" };

            Console.WriteLine($"Int value: {intBox.Value}");

            Console.WriteLine($"String value: {stringBox.Value}");

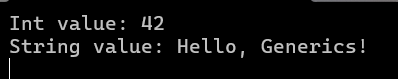
            Console.ReadLine();

        }

    }

}

**Output:**



**Conclusion:**

From this lab, we learned about generics. Hence, generics is created and the values are set and displayed.

# Lab 12: Implementation of Delegates

**Objective:**

* To implement delegates

**Source Code:**

using System;

namespace Chapter\_Four\_Delegate

{

    internal class Program

    {

        public delegate void Calculation(int a, int b);

        static void Add(int a, int b) => Console.WriteLine("Sum: " + (a + b));

        static void Sub(int a, int b) => Console.WriteLine("Difference: " + (a - b));

        static void Main(string[] args)

        {

            Calculation c = Add;

            c(10, 2);

            c = Sub;

            c(10, 5);

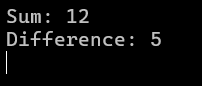
            Console.ReadLine();

        }

    }

}

**Output:**



**Conclusion:**

From this lab, we learned to implement delegates. Hence, a delegate Calculation is created and methods are assigned and results are displayed.

# Lab 13: Implementation of Lambda Expression

**Objective:**

* To implement Lambda Expressions

**Source Code:**

using System;

using System.Collections.Generic;

namespace Chapter\_Four\_Lambda

{

    internal class Program

    {

        static void Main(string[] args)

        {

            List<int> mylist = new List<int> { 1,2,3,4,5,6,7,8,9,10,11,12,13};

            Console.WriteLine("\nEven Numbers Using Lambda:");

            List<int>listeven=mylist.FindAll(x=>x%2==0);

            for(int i = 0;i<listeven.Count;i++)

                   Console.WriteLine(listeven[i]);

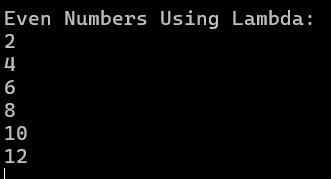
            Console.ReadLine();

        }

    }

}

**Output:**



**Conclusion:**

From this lab, we learned to implement lambda expression. Hence, a list is created with numbers and even numbers are displayed using lambda expression.

# Lab 14: Implementation of Exception Handling

**Objective:**

* To implement exception handling

**Source Code:**

using System;

namespace CHapter\_Four\_Try\_Catch

{

    internal class Program

    {

        static void Main(string[] args)

        {

            Console.WriteLine("Enter Number");

            try

            {

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

                int iNo = num / 2;

            }

            catch (Exception ex) {

                if (ex.GetType().ToString() == "System.FormatException")

                {

                    Console.WriteLine("Enter a numeric value.");

                }

            }

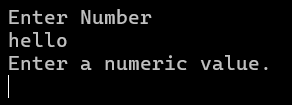
            Console.ReadLine();

        }

    }

}

**Output:**



## Conclusion:

From this program we learned to implement Exception Handling. Hence, the program was written to demonstrate the usage of Exception Handling.

# Lab 15: Implementation of LINQ

**Objective:**

* To implement LINQ

**Source Code:**

using System;

using System.Collections.Generic;

using System.Linq;

namespace LINQ

{

    public class Student

    {

        public string Name { get; set; }

        public string Address { get; set; }

    }

    internal class Program

    {

        public static void Main(string[] args)

        {

            List<Student> students = new List<Student>

            {

                new Student{Name="Ram",Address="Kirtipur"},

                new Student{Name="Hari",Address="Kathmandu"},

                new Student{Name="Sita",Address="Kirtipur"},

            };

            var res = students.Where(s => s.Address == "Kirtipur").ToList();

            foreach(var student in res)

            {

                Console.WriteLine($"Name:{student.Name},Address:{student.Address}");

            }

            Console.ReadLine();

        }

    }

}

**Output:**



# Lab 16: Implementation of Web Forms

**Objective:**

* To implement web forms

**Source Code:**

**PageCode.aspx:**

<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="PageCode.aspx.cs" Inherits="Basic\_Application\_Web\_Forms.PageCode" %>

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

     <div class="form-group">

    <label for="exampleInputNumber1">Number 1</label>

         <asp:RequiredFieldValidator runat="server" ControlToValidate="TextBox1" ErrorMessage="\*" />

    <asp:TextBox runat="server" class="form-control" id="TextBox1" />

  </div><br />

  <div class="form-group">

    <label for="exampleInputNumber2">Number 2</label>

    <asp:TextBox runat="server" class="form-control" id="TextBox2"  />

  </div>

    <br />

  <asp:Button runat="server" class="btn btn-primary" ID="Button1" Text="Submit" OnClick="Button1\_Click" /> <br /><br />

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

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

</asp:Content>

**PageCode.aspx.cs:**

using System;

namespace Basic\_Application\_Web\_Forms

{

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

    {

        protected void Button1\_Click(object sender, EventArgs e)

        {

            float f = Convert.ToInt32(TextBox1.Text);

            float g = Convert.ToInt32(TextBox2.Text);

            float h = f + g;

            Label1.Text = h.ToString();

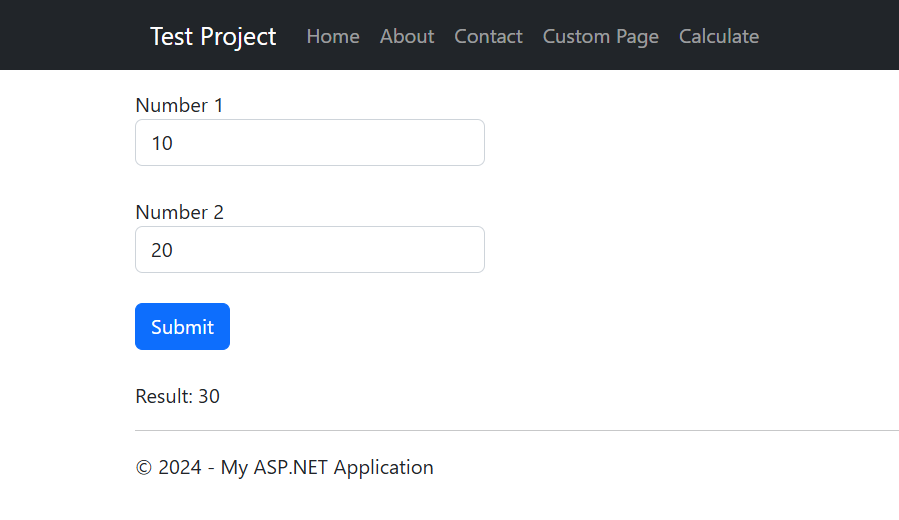
            Label1.Visible = true;

        }

    }

}

**Output:**

****

**Conclusion:**

From this lab, we learned to implement asp.NET web forms. Hence a webpage is created using web forms that takes two numbers and displays the result.

# Lab 17: Implementation of Database

**Objective:**

* To implement database

**Source Code:**

**Page.aspx:**

<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true" CodeBehind="ListPage.aspx.cs" Inherits="Basic\_Application\_Web\_Forms.ListPage" %>

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

    <div class="form-group">

        <label for="firstname">First Name</label>

        <asp:TextBox runat="server" class="form-control" ID="firstname" placeholder="Enter FirstName" />

    </div>

    <div class="form-group">

        <label for="lastname">Last Name</label>

        <asp:TextBox runat="server" class="form-control" ID="lastname" placeholder="Enter Lastname" />

    </div>

    <div class="form-group">

        <label for="address">Address</label>

        <asp:TextBox runat="server" class="form-control" ID="address" placeholder="Enter Address" />

    </div>

    <div class="form-group">

        <label for="phone">Phone</label>

        <asp:TextBox runat="server" class="form-control" ID="phone" placeholder="Enter Phone" />

    </div>

    <div class="form-group">

        <label for="email">Email</label>

        <asp:TextBox runat="server" class="form-control" ID="email" placeholder="Enter Email" />

    </div>

    <br />

    <asp:Button runat="server" class="btn btn-primary" ID="Button1" Text="Submit" OnClick="Button1\_Click" />

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

</asp:Content>

**Page.aspx.cs:**

using System;

using System.Configuration;

using System.Data.SqlClient;

namespace Basic\_Application\_Web\_Forms

{

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

    {

        SqlConnection \_conn;

        SqlCommand \_command;

        string strQuery, strCon;

        protected void Button1\_Click(object sender, EventArgs e)

        {

            strCon = ConfigurationManager.ConnectionStrings["ConnectDatabase"].ConnectionString;

            if (firstname.Text != "" && lastname.Text != "" && address.Text != "" && phone.Text != "" && email.Text != "")

            {

                \_conn = new SqlConnection(strCon);

                \_conn.Open();

                strQuery = "INSERT INTO Students (FirstName, LastName, Address, Phone, Email) " +

    "VALUES ('" + firstname.Text + "', '" + lastname.Text + "', '" + address.Text + "', '" + phone.Text + "', '" + email.Text + "')";

                \_command = new SqlCommand(strQuery, \_conn);

                int iRowAffect = \_command.ExecuteNonQuery();

                if (iRowAffect == 1)

                    Label1.Text = "Inserted Successfully";

                else

                    Label1.Text = "Unsuccesfully attempt";

                Label1.Visible = true;

                \_conn.Close();

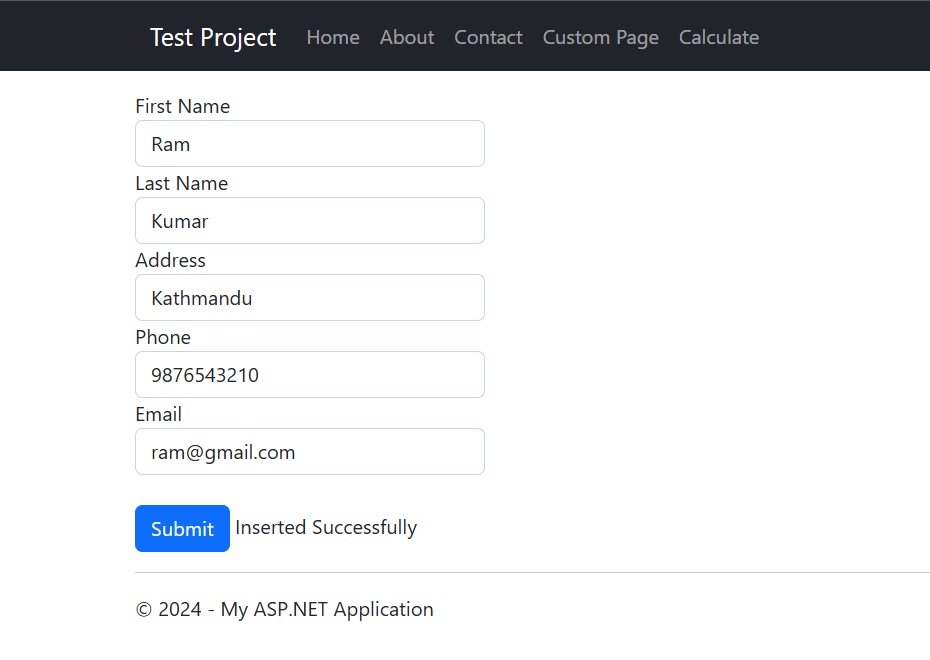
            }

        }

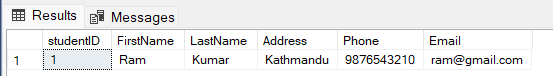
    }

}

**Output (Web Page):**

****

**Output (Database):**

****

**Conclusion:**

From this lab, we learned about database connectivity in asp.NET. Hence, a webpage is created using .NET web forms and a database STUDENT is connected and data is inserted and stored in the table and displayed.