1. **Describe the enumerations programming constructs, which provides a human-readable form of a series of related constant values in C#..**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace EnumerationDemo

{

class ProgramOne

{

enum Colors

{

Green,

Blue,

Yellow,

Violet,

Red,

Orange,

Pink

}

static void Main(string[] args)

{

foreach (var color in Enum.GetValues(typeof(Colors)))

{

Console.WriteLine("{0} : {1}", color, (int)color);

}

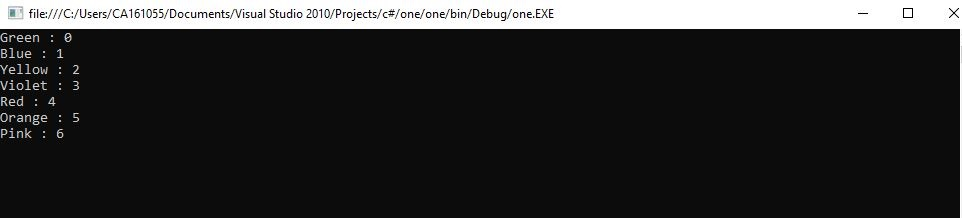
Console.Read();

}

}

}

**OUTPU**



1. **Check Whether the Entered Year is a Leap Year or Not**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ProgramTwo

{

class Program

{

static void Main(string[] args)

{

long year;

Console.WriteLine("Enter the year");

year = Int64.Parse(Console.ReadLine());

if (checkYear(year))

Console.WriteLine("{0} is a Leap year", year);

else

Console.WriteLine("{0} is not a Leap year", year);

Console.ReadLine();

}

static bool checkYear(long year)

{

if (year % 400 == 0)

return true;

else if (year % 100 == 0)

return false;

else if (year % 4 == 0)

return true;

else

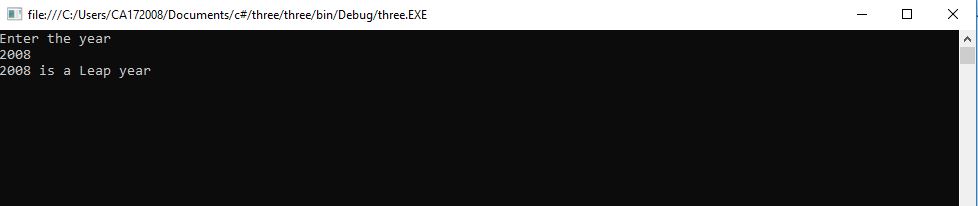
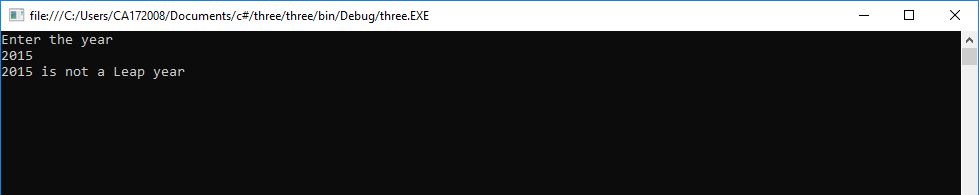
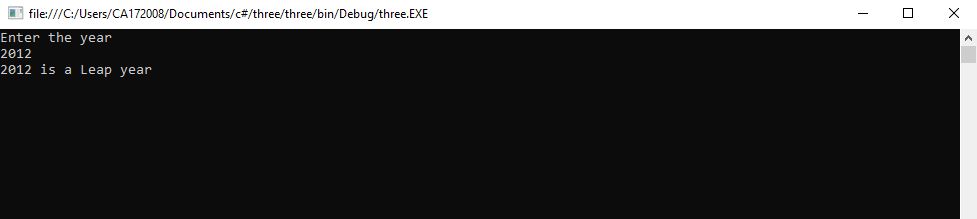
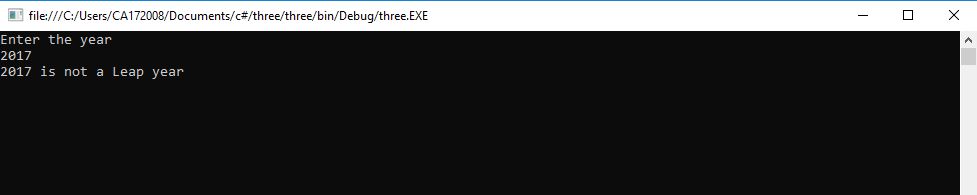
return false;

}

}

}

**OUTPUT**



1. **Program to display the addition using the windows application.**

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;

namespace three

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

int num1 = Int16.Parse(textBox1.Text);

int num2 = Int16.Parse(textBox2.Text);

int sum = num1 + num2;

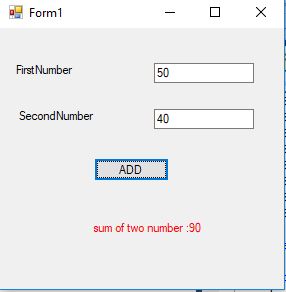
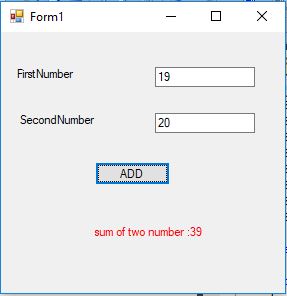
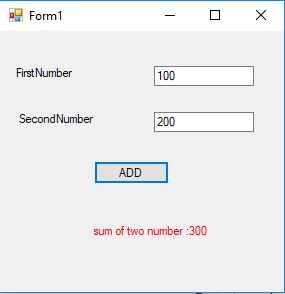
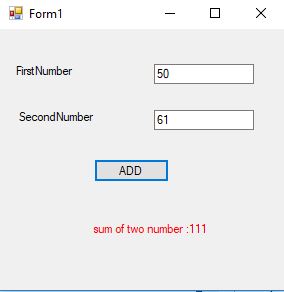
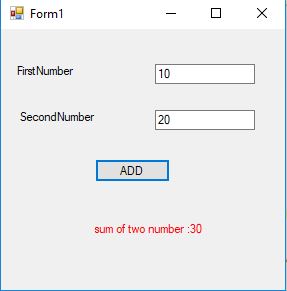
label3.Text = "sum of two number :" + sum;

}

}

}

**OUTPUT**



1. **Program to display the addition, subtraction, multiplication and division of two number using console applications.**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ProgramFive

{

class Program

{

static void Main(string[] args)

{

double num1, num2;

double sum, sub, mul, div;

Console.WriteLine("Enter the two numbers");

num1 = Double.Parse(Console.ReadLine());

num2 = Double.Parse(Console.ReadLine());

sum = num1 + num2;

sub = num1 - num2;

mul = num1 \* num2;

div = num1 / num2;

Console.WriteLine();

Console.WriteLine("----------------------------");

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

Console.WriteLine("Substraction: {0}", sub);

Console.WriteLine("Multiplication: {0}", mul);

Console.WriteLine("Division: {0}", div);

Console.WriteLine("----------------------------");

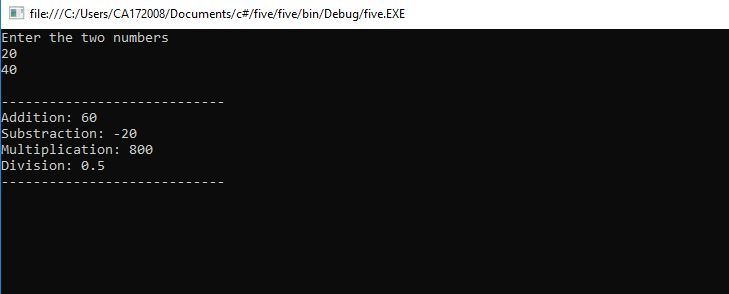
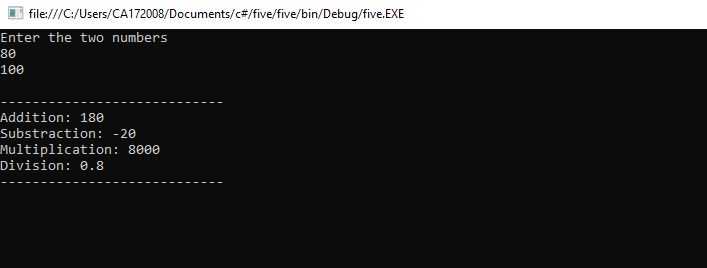
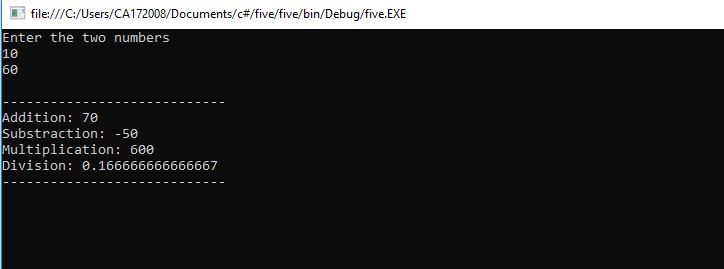
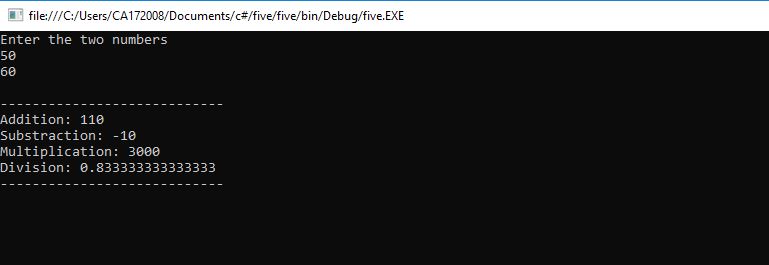
Console.ReadLine();

}

}

}

**OUTPUT**



1. **Program to display the first 10 natural numbers and their sum using console application**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ProgramSix

{

class Program

{

static void Main(string[] args)

{

int sum = 0;

Console.WriteLine("------------------");

Console.WriteLine("First 10 natural numbers");

Console.WriteLine("------------------");

for (int i = 1; i <= 10; i++)

{

sum += i;

Console.WriteLine(i);

}

Console.WriteLine("------------------");

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

Console.WriteLine("------------------");

Console.ReadLine();

}

}

}

**OUTPUT**



1. **Write a program to convert input string from lower to upper and upper to lower case.**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace LowUp

{

class Program

{

static void Main(string[] args)

{

string str1;

char[] arr1;

int l,i;

l=0;

char ch;

Console.Write("Input the string : ");

str1 = Console.ReadLine();

l=str1.Length;

arr1 = str1.ToCharArray(0, l);

Console.Write("\nAfter conversion, the string is : ");

for (i = 0; i < l; i++)

{

ch = arr1[i];

if (Char.IsLower(ch))

Console.Write(Char.ToUpper(ch));

else

Console.Write(Char.ToLower(ch));

}

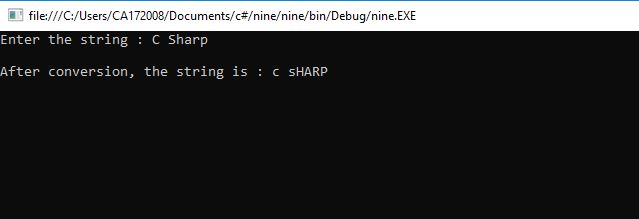
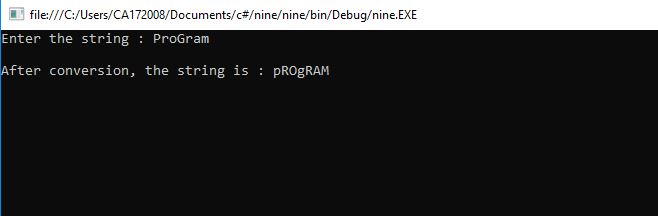
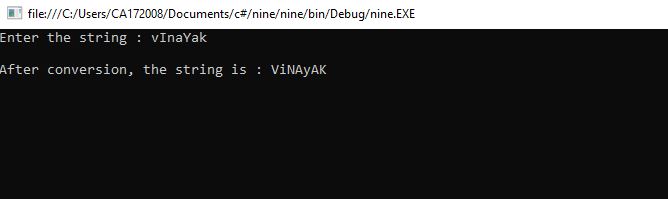
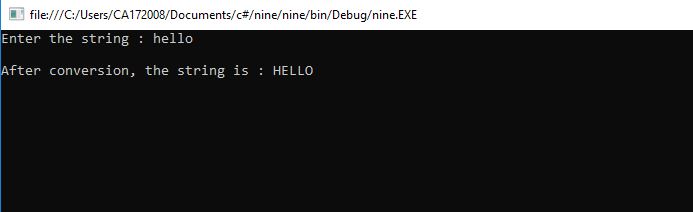
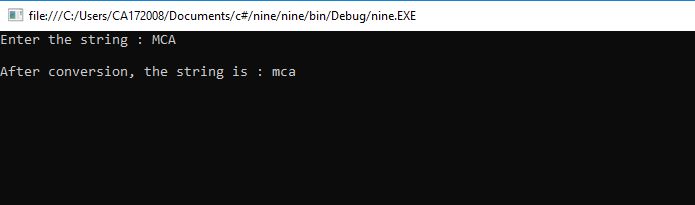
Console.ReadLine();

}

}

}

**OUTPUT**



1. **Demonstrate Command line arguments processing.**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Sixteen

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("\nNumber of CommadLine Arguments :" + args.Length);

Console.Write("\nCommandline Arguments Are :\t");

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

{

Console.Write(args[i] + "\t");

}

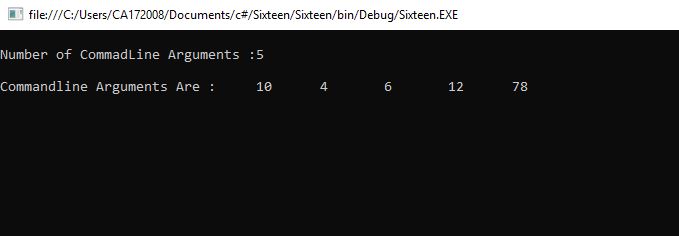
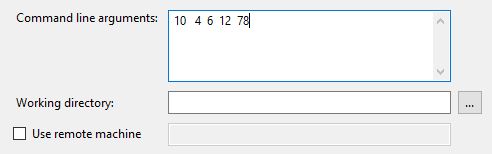
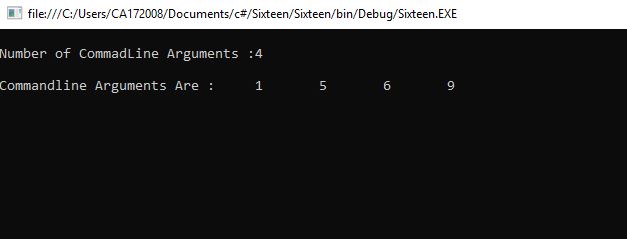
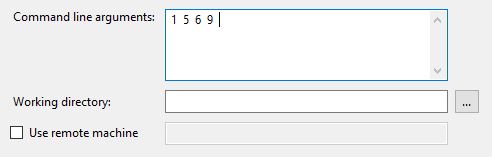
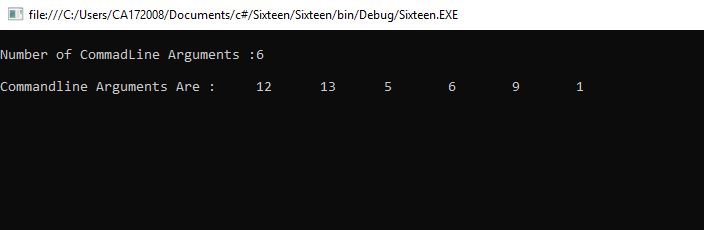
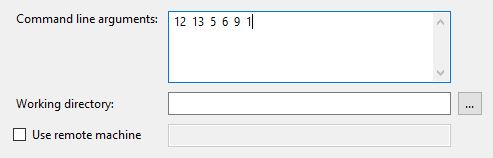
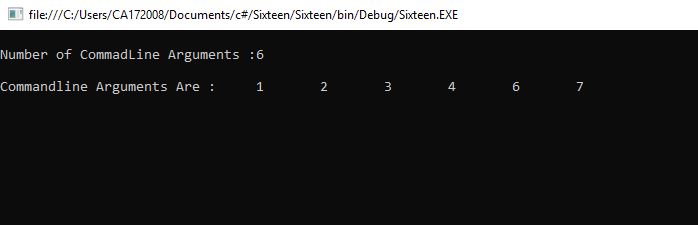
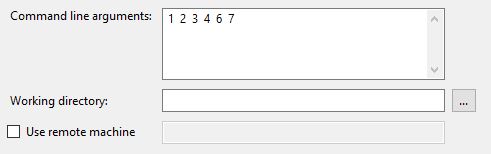
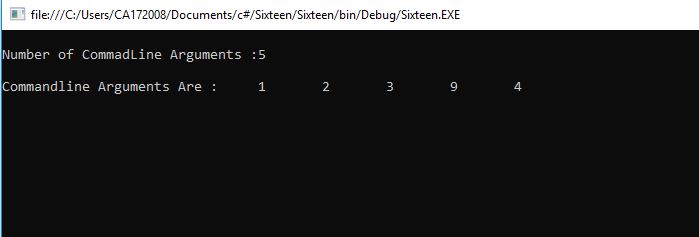
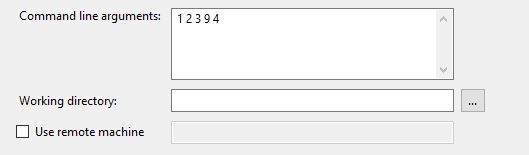
Console.ReadLine();

}

}

}

**OUTPUT**



1. **Perform operator overloading.**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

class Overload

{

Public int value;

public static overload operator +(overload a, overload b)

{

Overload overload = new Overload();

overload.value = a.value + b.value;

return overload;

}

public static overload operator ++(overload c)

{

c.value++;

return c;

}

}

class Program

{

static void Main()

{

Overload d = new Overload();

d++;

Console.WriteLine(d.value);

d++;

Console.WriteLine(d.value);

overload g = new overload();

g++;

Console.WriteLine(g.value);

overload t = d + g;

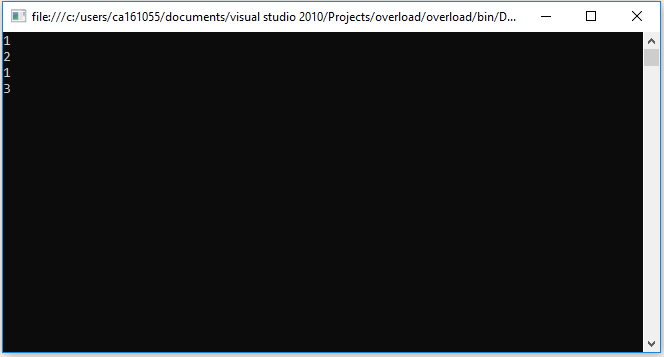
Console.WriteLine(t.value);

Console.ReadLine();

}

}

**OUTPUT**

****

1. **Find the second largest element in a single dimensional array.**

using System;

using System.Collections.Generic;

using System.Text;

namespace Secondlarge

{

class Program

{

static void Main(string[]args)

{

int n, i, temp;

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Console.WriteLine("Second largest element of Given Array!...");

Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

Console.WriteLine("Enter the size of an Array:");

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

Console.WriteLine("Enter the array elements:");

int[] arr = new int[n];

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

{

arr[i] = Convert.ToInt32(Console.ReadLine());

}

int l1, l2;

l1 = arr[0];

l2 = arr[1];

if (l1 < l2)

{

temp = l1;

l1 = l2;

l2 = temp;

}

for (i = 2; i < n; i++)

{

if (arr[i] > l2)

l2 = arr[i];

if (l2 > l1)

{

temp = l2;

l2 = l1;

l1 = temp;

}

}

Console.WriteLine("\n second largest element is : " + l2);

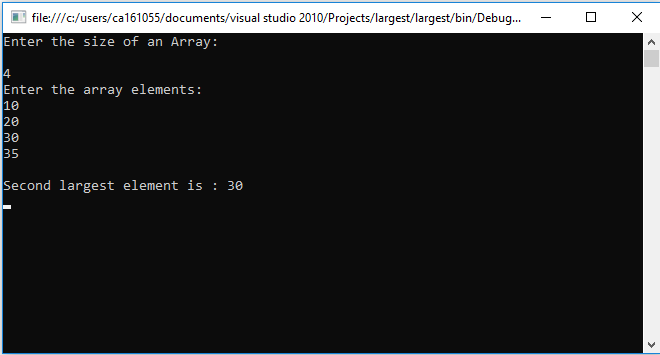
Console.ReadLine();

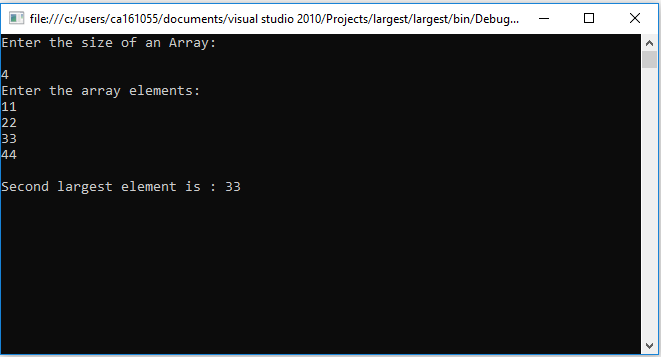
}

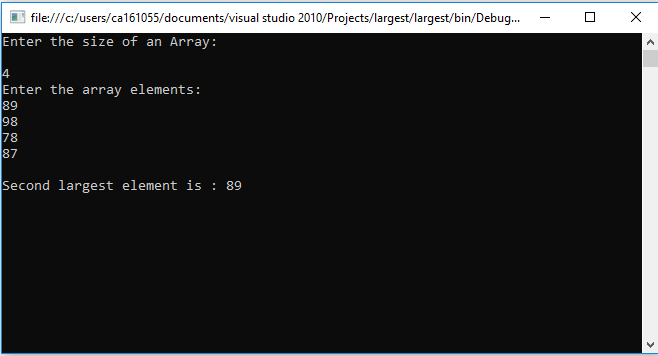
}

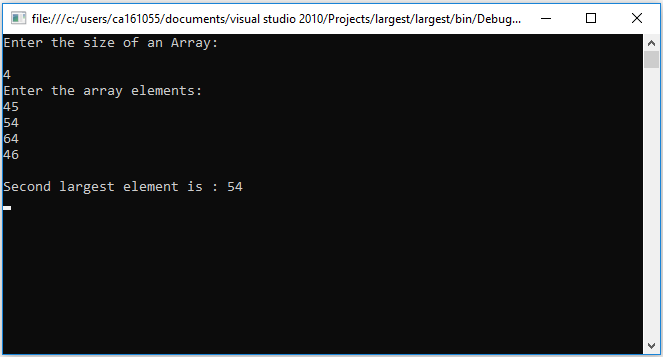
}

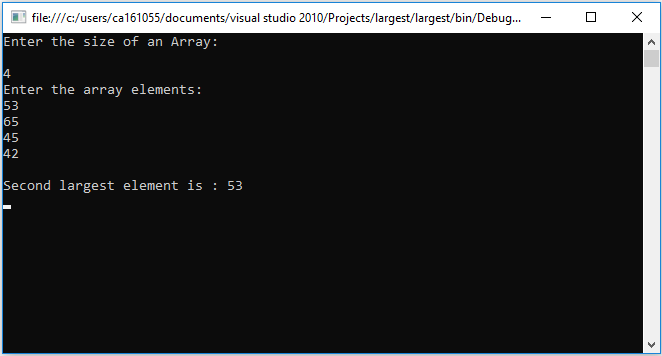
**OUTPUT**

****

****

****

****

****

1. **Describe the enumerations programming constructs, which provides a human-readable form of a series of related constant values in C#.**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication20

{

class Program

{

enum Month { Jan = 1, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec }

static void Main(string[] args)

{

for (int i = 0; i < 12; i++)

{

Console.WriteLine("{0}.{1}", (int)Month.Jan + i, Month.Jan + i);

}

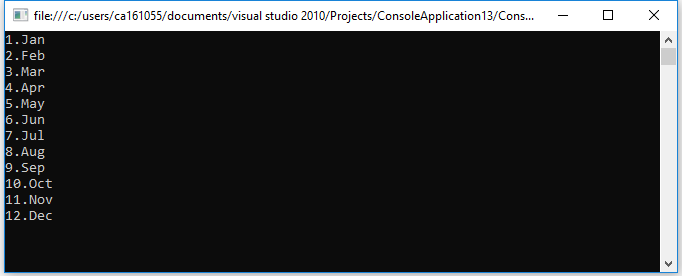
Console.ReadLine();

}

}

}

**OUTPUT**

****

1. **Create classes, they are reference types in C# and hence are allocated on the heap. Classes provide object-oriented constructs such as encapsulation, polymorphism, and inheritance. For instance, the program should print John. Doe twice, illustrating that objects are reference types, allocated on the heap implement the same using C#.**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace JoeDoe

{

public class UseClasses

{

public static void Main()

{

Person hs = new Person("Hitesh", "Seth");

Person jd = hs;

jd.FirstName = "John";

jd.LastName = "Doe";

Console.WriteLine(hs.GetFullName());

Console.WriteLine(jd.GetFullName());

Console.ReadLine();

}

}

public class Person

{

private string sFirstName, sLastName;

public Person(string FirstName, string LastName)

{

this.sFirstName = FirstName;

this.sLastName = LastName;

}

public string FirstName

{

get

{

return sFirstName;

}

set

{

sFirstName = value;

}

}

public string LastName

{

get

{

return sLastName;

}

set

{

sLastName = value;

}

}

public String GetFullName()

{

return this.FirstName + "." + this.LastName;

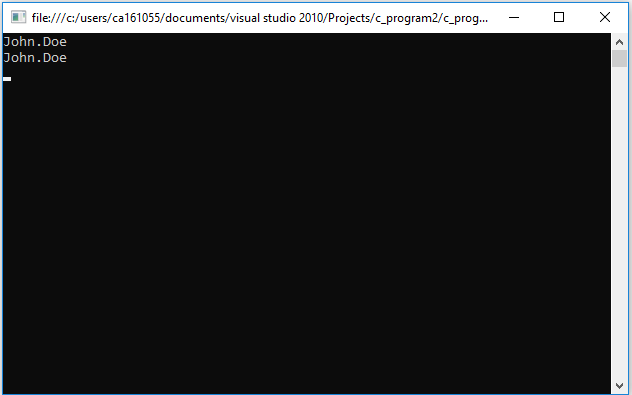
Console.ReadLine();

}

}

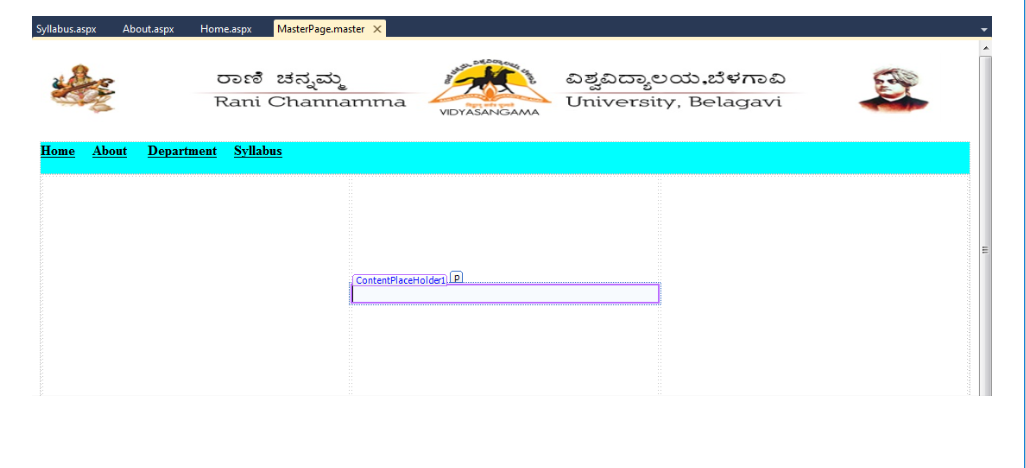
}

**OUTPUT**

****

1. **Work with Page using ASP.Net.**

**Master Page Design:**



<%@ Master Language="C#" AutoEventWireup="true" CodeFile="MasterPage.master.cs" Inherits="MasterPage" %>

<!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>

    <img src="rcub1.jpg" style="width: 1043px; height: 80px" />

</head>

<body>

    <form id="form1" runat="server">

    <div style="background-color: #00FFFF; height: 37px;">

        <asp:HyperLink ID="HyperLink1" runat="server"

            style="color: #000000; font-weight: 700" NavigateUrl="~/Home.aspx">Home</asp:HyperLink>

&nbsp;&nbsp;&nbsp;&nbsp;

        <asp:HyperLink ID="HyperLink2" runat="server"

            style="color: #000000; font-weight: 700" NavigateUrl="~/About.aspx">About</asp:HyperLink>

&nbsp;&nbsp;

        &nbsp;&nbsp;

        <asp:HyperLink ID="HyperLink3" runat="server"

            style="color: #000000; font-weight: 700" NavigateUrl="~/Department.aspx">Department</asp:HyperLink>

&nbsp;&nbsp;&nbsp;&nbsp;

        <asp:HyperLink ID="HyperLink4" runat="server"

            style="color: #000000; font-weight: 700" NavigateUrl="~/Syllabus.aspx">Syllabus</asp:HyperLink>

&nbsp;&nbsp;&nbsp;&nbsp;

        </div>

    <div>

        <table style="width: 100%; height: 278px;">

            <tr>

                <td>

                    &nbsp;</td>

                <td>

                    <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">

                        <p>

                            &nbsp;</p>

                    </asp:ContentPlaceHolder>

                </td>

                <td>

                    &nbsp;</td>

            </tr>

        </table>

    </div>

    <div style="height: 28px; background-color: #00FFFF">

     &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;<strong>&nbsp;Copyright Information Rcub.com</strong></div> </form> </body></html>

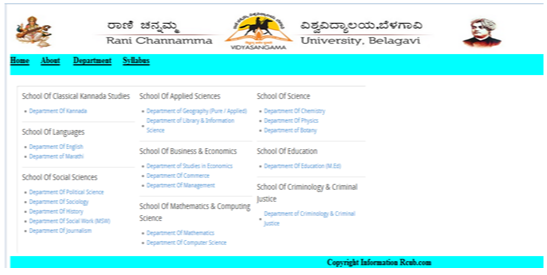
**OUTPUT**

**Home:**

**About:**

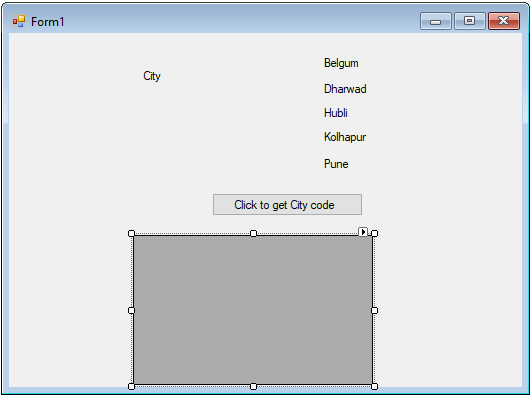


**Department:**



**Syllabus:** 

1. **Describe access data source through ADO.NET.**

****

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;

namespace DataAccessADO

{

public partial class Form1 : Form

{

DataSet ds = new DataSet();

SqlConnection cs = new SqlConnection("Data Source=CS-05; Initial Catalog=City; user id=sa;password=Password@123;");

SqlDataAdapter da = new SqlDataAdapter();

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

da.SelectCommand = new SqlCommand("SELECT \* FROM t1", cs);

ds.Clear();

da.Fill(ds);

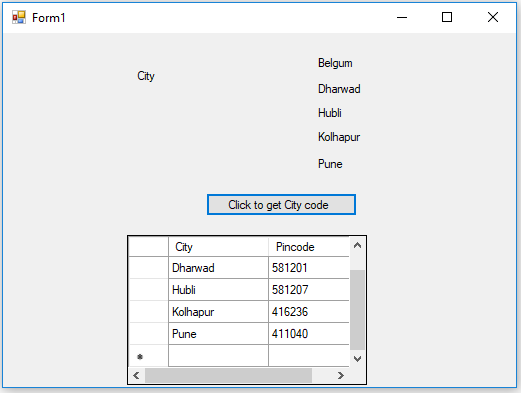
dgv.DataSource = ds.Tables[0];

}

}

}

**OUTPUT**

****

1. **Describe delegates, events, errors and exceptions.**

using System;

using System.Collections.Generic;

using System.Text;

namespace Delegates

{

public delegate void devidezero(int n);

class EventProgram

{

event devidezero MyEvent;

public EventProgram(int n)

{

try

{

Decimal dresult = Decimal.Divide(n, 0);

Console.WriteLine("Result is :{0}", dresult);

}

catch (DivideByZeroException exdiv)

{

Console.WriteLine("Caught Divide By Zero Exception : {0}", exdiv.Message);

}

catch (Exception ex)

{

Console.WriteLine("Caught Exception : {0}", ex.Message);

}

finally

{

Console.WriteLine("In Finally");

Console.ReadLine();

}

}

static void Main(string[] args)

{

int n;

Console.WriteLine("Enter the value of n:");

n = Convert.ToUInt16(Console.ReadLine());

EventProgram obj1 = new EventProgram(n);

obj1.MyEvent(n);

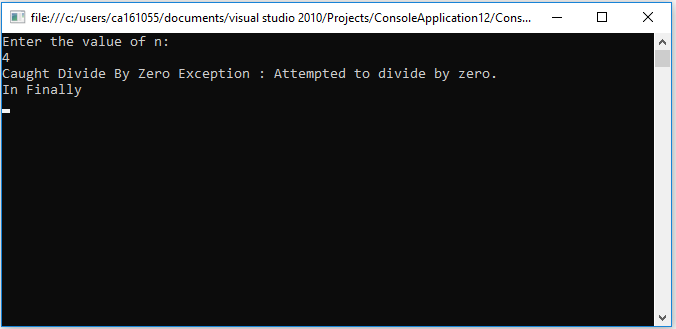
Console.WriteLine();

}

}

}

**OUTPUT**

****

1. **Program to illustrate the use of different properties in C#.**

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;

namespace Properties

{

public partial class Form1 : Form

{

private double seconds;

public double Hours

{

get { return seconds; }

set

{

if (value < 0 || value > 24)

throw new ArgumentOutOfRangeException(

"{nameof(value)} must be between 0 and 24.");

seconds = value \* 3600;

}

}

public Form1()

{

InitializeComponent();

this.StartPosition = FormStartPosition.CenterScreen;

this.Size = new Size(400, 400);

this.FormBorderStyle = FormBorderStyle.FixedToolWindow;

this.MaximizeBox = false;

}

private void button1\_Click(object sender, EventArgs e)

{

Form1 t = new Form1();

t.Hours = Convert.ToDouble(textBox1.Text);

textBox2.Text = Convert.ToString(t.Hours);

}

private void button2\_Click(object sender, EventArgs e)

{

this.BackColor = Color.Red;

this.Font = new Font("Georgia", 16);

}

private void button3\_Click(object sender, EventArgs e)

{

this.BackColor = Color.Pink;

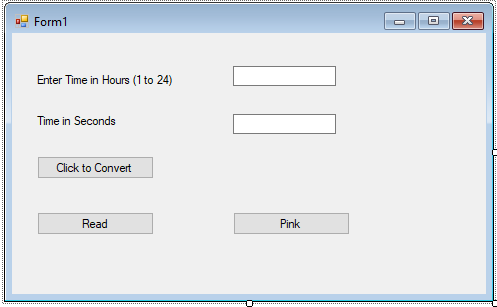
this.Font = new Font("Times New Roman", 25);

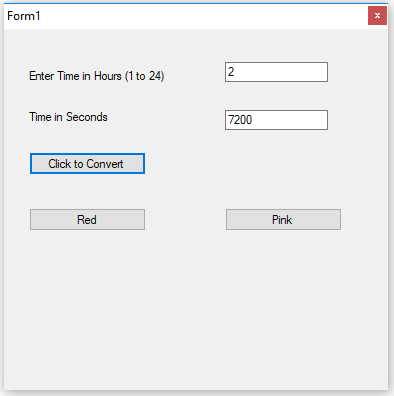
}

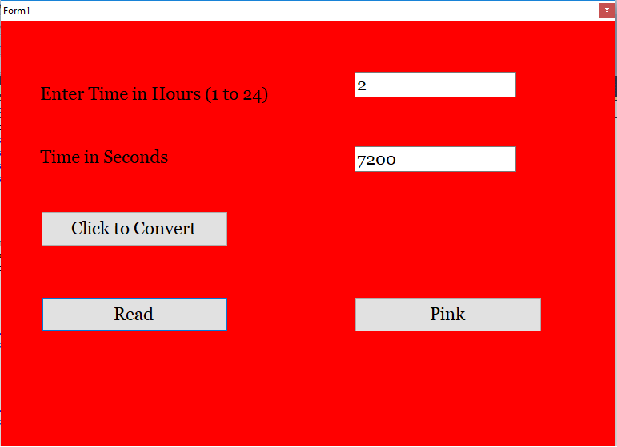
}

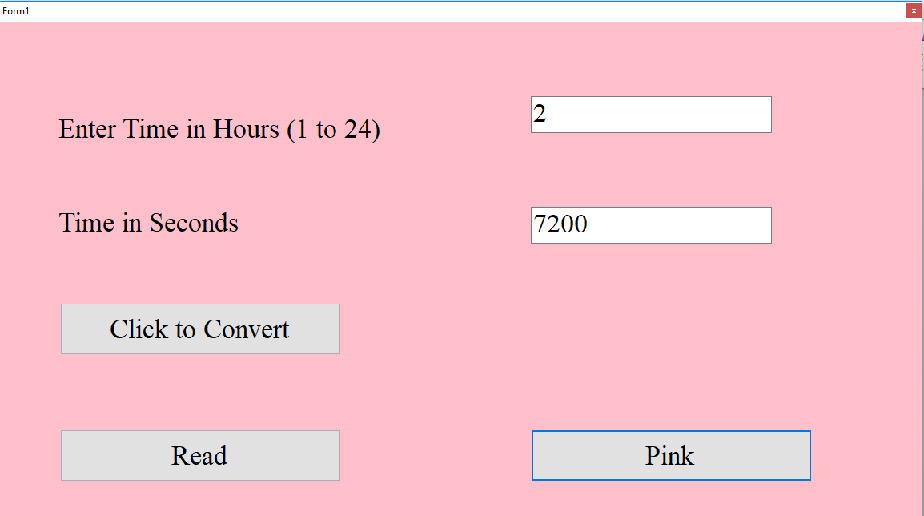
}

**OUTPUT**

****

****

****

****

1. **Demonstrate Command line arguments processing.**

**cmd.cs**

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System;

using System.IO;

using System.Threading.Tasks;

namespace cmd

{

class Program

{

public static void Main(string[] args)

{

String path;

path = args[0];

String lines;

String lines1;

int n;

lines = File.ReadAllText(path);

Console.WriteLine(lines);

lines1 = lines.ToUpper();

Console.WriteLine("After Conversion");

Console.WriteLine(lines1);

Console.ReadLine();

Console.Writeline("Press enter to get total number of lines");

n = File.ReadLines(path).Count();

Console.WriteLine("Number of lines :" +n);

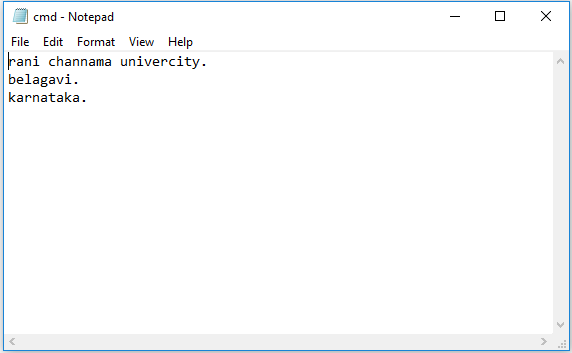
Console.ReadKey();

}

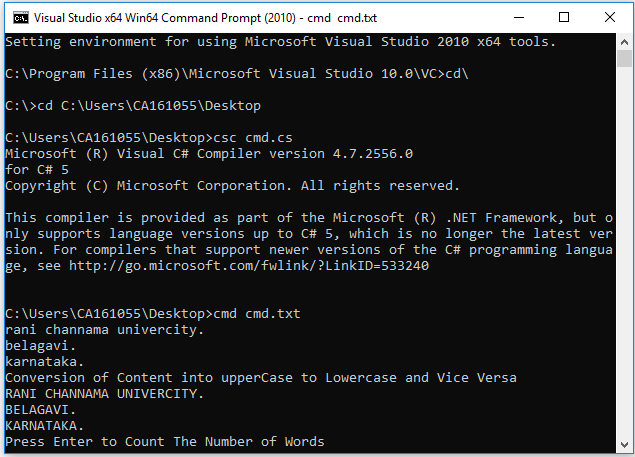
}

}

**cmd.txt**

****

**OUTPUT**

****

1. **Program to multiply to matrices using Rectangular arrays.**

using System;

using System.Collections;

using System.Data;

public class MatrixMul

{

public static void Main()

{

int i, j, k, r1, c1, r2, c2, sum = 0;

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

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

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

Console.Write("\n\nMultiplication of two Matrices\n");

Console.Write("----------------------------------\n");

Console.Write("\nInput the number of rows and columns of the first matrix :\n");

Console.Write("Rows : ");

r1 = Convert.ToInt32(Console.ReadLine());

Console.Write("Columns : ");

c1 = Convert.ToInt32(Console.ReadLine());

Console.Write("\nInput the number of rows of the second matrix :\n");

Console.Write("Rows : ");

r2 = Convert.ToInt32(Console.ReadLine());

Console.Write("Columns : ");

c2 = Convert.ToInt32(Console.ReadLine());

if (r1 != c1 && r2 != c2)

{

Console.Write("This Is Rectangular Matrix");

if (c1 != r2)

{

Console.Write("Mutiplication of Matrix is not possible.");

Console.Write("\nColumn of first matrix and row of second matrix must be same.");

}

else

{

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

for (i = 0; i < r1; i++)

{

for (j = 0; j < c1; j++)

{

Console.Write("element - [{0}],[{1}] : ", i, j);

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

}

}

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

for (i = 0; i < r2; i++)

{

for (j = 0; j < c2; j++)

{

Console.Write("element - [{0}],[{1}] : ", i, j);

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

}

}

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

for (i = 0; i < r1; i++)

{

Console.Write("\n");

for (j = 0; j < c1; j++)

Console.Write("{0}\t", arr1[i, j]);

}

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

for (i = 0; i < r2; i++)

{

Console.Write("\n");

for (j = 0; j < c2; j++)

Console.Write("{0}\t", brr1[i, j]);

}

//multiplication of matrix

for (i = 0; i < r1; i++)

for (j = 0; j < c2; j++)

crr1[i, j] = 0;

for (i = 0; i < r1; i++) //row of first matrix

{

for (j = 0; j < c2; j++) //column of second matrix

{

sum = 0;

for (k = 0; k < c1; k++)

sum = sum + arr1[i, k] \* brr1[k, j];

crr1[i, j] = sum;

}

}

Console.Write("\nThe multiplication of two matrix is : \n");

for (i = 0; i < r1; i++)

{

Console.Write("\n");

for (j = 0; j < c2; j++)

{

Console.Write("{0}\t", crr1[i, j]);

}

}

}

}

else

{

Console.WriteLine("This is not a rectangular matrix");

}

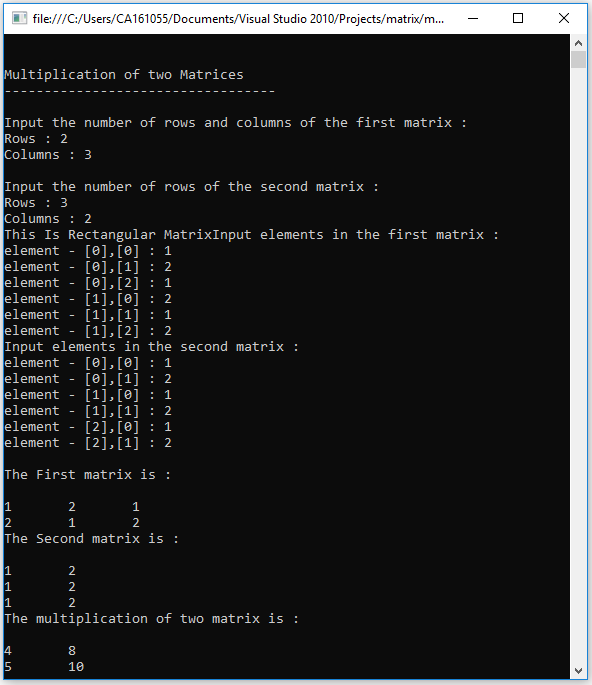
Console.ReadLine();

Console.Write("\n\n");

}

}

**OUTPUT**

****

1. **Use of Virtual and override keyword in C# with a simple Program.**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace LastPrg

{

class A

{

public virtual void show()

{

Console.WriteLine("Hello: Base Class!");

Console.ReadLine();

}

}

class B : A

{

public override void show()

{

Console.WriteLine("Hello: Derived Class!");

Console.ReadLine();

}

}

class Polymorphism

{

public static void Main()

{A a1 = new A();

a1.show();

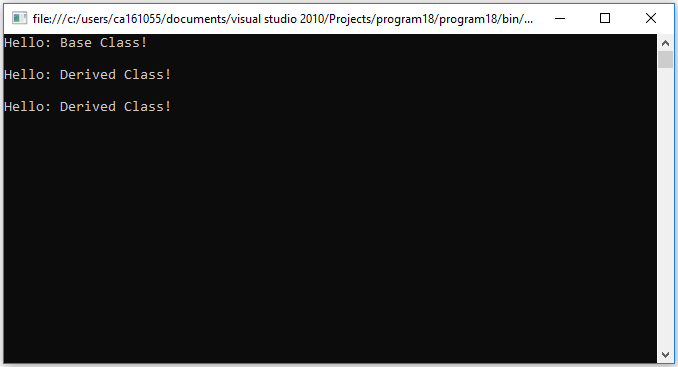
B b1 = new B();

b1.show();

A a2 = new B();

a2.show();}}}

**OUTPUT**

****