1. Write a program to find the factorial of given number using the concept of delegate

Factorial.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace factorial

{

public class Factorial

{

public int calculatefactorial(int n)

{

int fact = 1;

if( n < 0)

{

Console.WriteLine("Error! Factorial of a negative number does not exist.");

}

else

{

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

{

fact \*= i;

}

}

return fact;

}

}

}

Program.cs

using System;

using factorial;

namespace Lab2

{

internal class Program

{

public delegate int calcfactorial(int x);

static void Main(string[] args)

{

Factorial factorialcalc = new Factorial();

calcfactorial calculatefactorial = new calcfactorial(factorialcalc.calculatefactorial);

Console.WriteLine("Enter an integer:");

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

int result = calculatefactorial.Invoke(input);

Console.WriteLine($"The factorial of {input} is {result}");

}

}

}

1. Create a Class Quadratic having a, b, c, x1, and x2 as instance variables. Create a function name Input (no parameter and no return type) to take the user input for a, b, and c. Then create another function name double[ ] Calculate ( ) to calculate two roots and assign to variable x1 and x2 and return these two roots must return to main function. Create another class Imain having main () function to create an object of Quadratic class and invoke the function.
2. Create a class Student having instance variable age and name and class also contains a function name void input( ) for user input age and name. Then create another class Imain and create an array of size 5 of Student then store the Student in array and print those records of array whose age is greater than or equal to 24.

Pseudocode:

class Student

{

int age;

void input( )

{

//input goes here

}

}

class Imain

{

Student[ ] lstStudent = new Student[ 5];

//Create 5 objects of Student and store in Array lstStudent

//print the roll and name of students stored in array whose ages is greater or equal to 24

}

1. Create a class Time with three integer member variables hr ,min,sec. The class also will contain the method Tim Sum(Time t1,Time t2) method that will return the sum of t1and t2 Create a class TimeDemo with main method that will create an object of Time and to invoke the Sum function and print the added time

Output :

t1—>5:40:40

t2—>4:40:50

Total Sum t3—>10:21:30

1. Write a program to read a file from input.txt then copy the content of input.txt to output.txt.

E.g

**//input.txt**

Hello and Welcome

It is the first content

of the text file from first file

**output.txt:**

Hello and Welcome

It is the first content

of the text file from first file

1. Write a program to read a file from Input.txt then display those words which ends with the character ‘g’.

E.g

**//Input.txt**

C# is a OOP Programming Language

Mr Hari Gurung loves a programming

**Output**

Programming

Gurung

Programming

1. Create an interface called ICalculator which has methods int add(int x ,int y) and int diff(int x,int y) to perform addition and subtraction of numbers passed as arguments. Then define a class that implements interface ICalculator.
2. Create an interface Shape which method get() and display().Create two classes Rectangle and Square which implements this interface.defines the member variable of these classes as per requirement in class itself.Create some instances of Rectangle and Square classes and demonstrate interface implementation by classes .
3. Create an abstract class called Vehicle with an abstract method startEngine() and a non-abstract method stopEngine(). Derive a concrete class Airplane from Vehicle, having instance variables such as model, manufacturer, and maxCapacity. The Airplane class should implement the abstract startEngine() method with appropriate logic for starting an aircraft engine. Additionally, provide concrete implementations for methods like takeOff() and land().

In your main class, create an instance of the Airplane class and demonstrate the usage of the inherited methods (startEngine(), stopEngine(), takeOff(), and land()) along with accessing the instance variables.

1. Create a class Employee which contains the properties like ID, Name, Address and Salary. Create another class Imain then print all the employees whose salary is greater than 40000 using the concept of Linq Query.
2. Create an array of Integer of size n then enter the element in array and find the sum of odd numbers of array elements using the concept of Lambda Expression.