using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SMSProject

{

class Program

{

static void Main(string[] args)

{

int[] num = new int[10];

int i = 0;

Console.WriteLine("Enter No");

string inp = Console.ReadLine();

foreach (var sn in inp.Split(' '))

{

num[i]= Convert.ToInt32(sn);

i++;

// work with n

}

for (i = 0; i < num.Length; i++)

{

Console.WriteLine(num[i]);

}

}

}

}

Console.ReadLine();

Console.Read();

Console.ReadKey();

using System;

class A

{

public static void Main()

{

int c = 0;

Console.WriteLine("The series is:");

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

{

c = c + i;

Console.Write(c + " ");

}

Console.Write("\nPress 'Enter' to exit the process...");

// another use of "Console.ReadKey()" method

// here it asks to press the enter key to exit

while (Console.ReadKey().Key != ConsoleKey.Enter)

{

Console.WriteLine("You pressed Enter Key");

}

}

}

 public static void Main()

    {

        // "DateTime" is a inbuilt class

        // for date and time

        DateTime d = DateTime.Now;

        // print the system date and time

        Console.WriteLine("System date: {0:d}\n"+

                        "System time: {0:t}", d);

        Console.Write("Press 'E' to exit the process...");

        // here it ask to press "E" to exit

        while (Console.ReadKey().Key != ConsoleKey.E) {

        }

    }

Student : rn , name

Sports : sports name , score

Test : m1, m2 , m3

Result : Display the result of the students

FinalScore = m1 + m2 + m3 + score

Dipslay all the details in Result class

using System;

// protected Access specifier

class student

{

int rn;

string name;

public void GetDetails()

{

Console.WriteLine("Enter RollNo");

rn = Convert.ToByte(Console.ReadLine());

Console.WriteLine("Enter Name");

name = Console.ReadLine();

}

public void DisplayDetails()

{

Console.WriteLine("RollNo is " + rn);

Console.WriteLine("Name is " + name);

}

}

class sports : student

{

string sportsName;

protected int score;

public void GetDetails()

{

base.GetDetails();

Console.WriteLine("Enter Sports Name");

sportsName = Console.ReadLine();

Console.WriteLine("Enter Score");

score = Convert.ToByte(Console.ReadLine());

}

public void DisplayDetails()

{

base.DisplayDetails();

Console.WriteLine("Sports Name is " + sportsName);

Console.WriteLine("Score is " + score);

}

}

class test : sports

{

protected int m1, m2, m3;

public void GetDetails()

{

base.GetDetails();

Console.WriteLine("Enter Marks 1");

m1 = Convert.ToByte(Console.ReadLine());

Console.WriteLine("Enter Marks 2");

m2 = Convert.ToByte(Console.ReadLine());

Console.WriteLine("Enter Marks 3");

m3 = Convert.ToByte(Console.ReadLine());

}

public void DisplayDetails()

{

base.DisplayDetails();

Console.WriteLine("Marks are " + m1 + " " + m2 + " and " + m3 );

}

}

class result : test

{

int FinalScore;

public void CalculateFinalScore()

{

GetDetails();

FinalScore = m1 + m2 + m3 + score;

}

public void DisplayDetails()

{

base.DisplayDetails();

Console.WriteLine("Your Final Score is " + FinalScore);

}

}

public class Program

{

static void Main()

{

result re = new result();

re.CalculateFinalScore();

re.DisplayDetails();

}

}

Q. Calculate Area of Square Formula used : area = side \* side

Q. Calculate Area of Triangle Formula used : area = ½ \* base \* height

Q. Calculate Area of Rectangle Formula used : area = length \* breadth

ALL OF YOU WRITE THESE PROGRAMS.

Acc to ankit k

Square class is base class of triangle and rectangle class

Shubham : APprocah was completely incorrect

Prasad, Harshit : They were using Procedural Approach

Parth : Correct in Apparoach

from 896899 -Parth Kulkarni to everyone: 11:42 AM

namespace Assignment1

{

class Program

{ Main()

{

}

}

class Square

{

public int GetArea(int a)

{

return a \* a;

}

}

class Triangle

{

int b, h;

public void GetValues()

{

Console.WriteLine("Enter base of triangle");

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

Console.WriteLine("Enter height");

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

}

public double DisplayArea()

{

return 0.5 \* b \* h;

}

}

class Rectangle

{

int l, b;

public void GetValues()

{

Console.WriteLine("Enter Length");

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

Console.WriteLine("Enter Breadth");

b = Convert.ToInt32(Console.ReadLine

**Without using Abstract class**

using System;

class square

{

float side;

float area;

public void GetValues()

{

Console.WriteLine("Enter Side");

side = float.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = side \* side;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

class rectangle

{

float length, breadth;

float area;

public void GetValues()

{

Console.WriteLine("Enter Length");

length = float.Parse(Console.ReadLine());

Console.WriteLine("Enter Breadth");

breadth = float.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = length \* breadth;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

class triangle

{

float Base, height;

float area;

public void GetValues()

{

Console.WriteLine("Enter Length");

Base = float.Parse(Console.ReadLine());

Console.WriteLine("Enter Breadth");

height = float.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = Base \* height;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

class Program

{

static void Main()

{

square square = new square();

square.GetValues();

square.CalculateArea();

square.DisplayArea();

rectangle rectangle = new rectangle();

rectangle.GetValues();

rectangle.CalculateArea();

rectangle.DisplayArea();

triangle triangle = new triangle();

triangle.GetValues();

triangle.CalculateArea();

triangle.DisplayArea();

}

}

abstract class figure

{

float dimension1, dimension2, area;

public abstract void GetValues();

public abstract void CalculateArea();

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

Abstract class : A class in which some methods are defined (concrete methods) and some of the methods are not defined (abstract methods)

It becomes base class for many classes

It can only be inherited

We cannot create object of abstract class

using System;

abstract class figure

{

protected float dimension1, dimension2, area;

public abstract void GetValues();

public abstract void CalculateArea();

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

class square : figure

{

public override void GetValues()

{

Console.WriteLine("Enter Side");

dimension1 = float.Parse(Console.ReadLine());

}

public override void CalculateArea()

{

area = dimension1 \* dimension2;

}

}

class rectangle : figure

{

public override void GetValues()

{

Console.WriteLine("Enter Length");

dimension1 = float.Parse(Console.ReadLine());

Console.WriteLine("Enter Breadth");

dimension2 = float.Parse(Console.ReadLine());

}

public override void CalculateArea()

{

area = dimension1 \* dimension2;

}

}

class triangle : figure

{

public override void GetValues()

{

Console.WriteLine("Enter Length");

dimension1 = float.Parse(Console.ReadLine());

Console.WriteLine("Enter Breadth");

dimension2 = float.Parse(Console.ReadLine());

}

public override void CalculateArea()

{

area = dimension1 \* dimension2;

}

}

class Program

{

static void Main()

{

square square = new square();

square.GetValues();

square.CalculateArea();

square.DisplayArea();

rectangle rectangle = new rectangle();

rectangle.GetValues();

rectangle.CalculateArea();

rectangle.DisplayArea();

triangle triangle = new triangle();

triangle.GetValues();

triangle.CalculateArea();

triangle.DisplayArea();

}

}

-------------------------------------------------------

Interface : Syntactical contract which contains only method declarations

These are just like Rules set by a company

Rules are followed by employees

using System;

interface figure

{

void GetValues();

void CalculateArea();

void DisplayArea();

}

class square : figure

{

float side , area;

public void CalculateArea()

{

area = side \* side;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

public void GetValues()

{

Console.WriteLine("Enter Side");

side = float.Parse(Console.ReadLine());

}

}

class rectangle : figure

{

float lenght, breadth, area;

public void GetValues()

{

Console.WriteLine("Enter Length");

lenght = float.Parse(Console.ReadLine());

Console.WriteLine("Enter Breadth");

breadth = float.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = lenght \* breadth;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

class triangle : figure

{

float Base , height , area;

public void GetValues()

{

Console.WriteLine("Enter Length");

Base = float.Parse(Console.ReadLine());

Console.WriteLine("Enter Breadth");

height = float.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = Base \* height;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

class Program

{

static void Main()

{

square square = new square();

square.GetValues();

square.CalculateArea();

square.DisplayArea();

rectangle rectangle = new rectangle();

rectangle.GetValues();

rectangle.CalculateArea();

rectangle.DisplayArea();

triangle triangle = new triangle();

triangle.GetValues();

triangle.CalculateArea();

triangle.DisplayArea();

}

}

Multiple Inheritance through Interface

using System;

interface displayArea

{

void DisplayArea();

}

interface figure

{

void GetValues();

void CalculateArea();

}

class square : figure , displayArea

{

float side , area;

public void CalculateArea()

{

area = side \* side;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

public void GetValues()

{

Console.WriteLine("Enter Side");

side = float.Parse(Console.ReadLine());

}

}

class rectangle : figure

{

float lenght, breadth, area;

public void GetValues()

{

Console.WriteLine("Enter Length");

lenght = float.Parse(Console.ReadLine());

Console.WriteLine("Enter Breadth");

breadth = float.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = lenght \* breadth;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

class triangle : figure

{

float Base , height , area;

public void GetValues()

{

Console.WriteLine("Enter Length");

Base = float.Parse(Console.ReadLine());

Console.WriteLine("Enter Breadth");

height = float.Parse(Console.ReadLine());

}

public void CalculateArea()

{

area = Base \* height;

}

public void DisplayArea()

{

Console.WriteLine("Area is " + area);

}

}

class Program

{

static void Main()

{

square square = new square();

square.GetValues();

square.CalculateArea();

square.DisplayArea();

rectangle rectangle = new rectangle();

rectangle.GetValues();

rectangle.CalculateArea();

rectangle.DisplayArea();

triangle triangle = new triangle();

triangle.GetValues();

triangle.CalculateArea();

triangle.DisplayArea();

}

}