**Лабораторная работа №13. Часть 2**

*«Классы. Интерфейсы»*

**Рекомендации по программированию**

* Поля предпочтительнее делать private.
* Интерфейс должен быть полным, т.е. предоставлять возможность выполнять любые разумные действия с классом. И одновременно минимально необходимым, т.е. без дублирования и пересечения возможностей методов.
* Каждый метод класса должен решать только одну задачу. Не стоит объединять два коротких независимых фрагмента кода в один метод.
* Если действия встречаются хотя бы дважды, стоит оформить их в отдельные функции.
* Все заданные методы, поля и экземпляры классов должны иметь осмысленные имена.
* Каждое имя интерфейса должно начинаться с буквы I.

**Требования к лабораторной**

* Каждый разрабатываемый класс должен, как правило, содержать следующие элементы: поля с различными спецификаторами, конструкторы с параметрами и без параметров, методы, свойства.
* Методы и свойства должны обеспечивать непротиворечивый, полный, минимальный и удобный интерфейс класса.
* При возникновении ошибок должны выбрасываться исключения
* К каждому отчету, помимо формулировки задания и скрина консоли, нужно прикрепить еще диаграмму классов

**Постановка задачи**

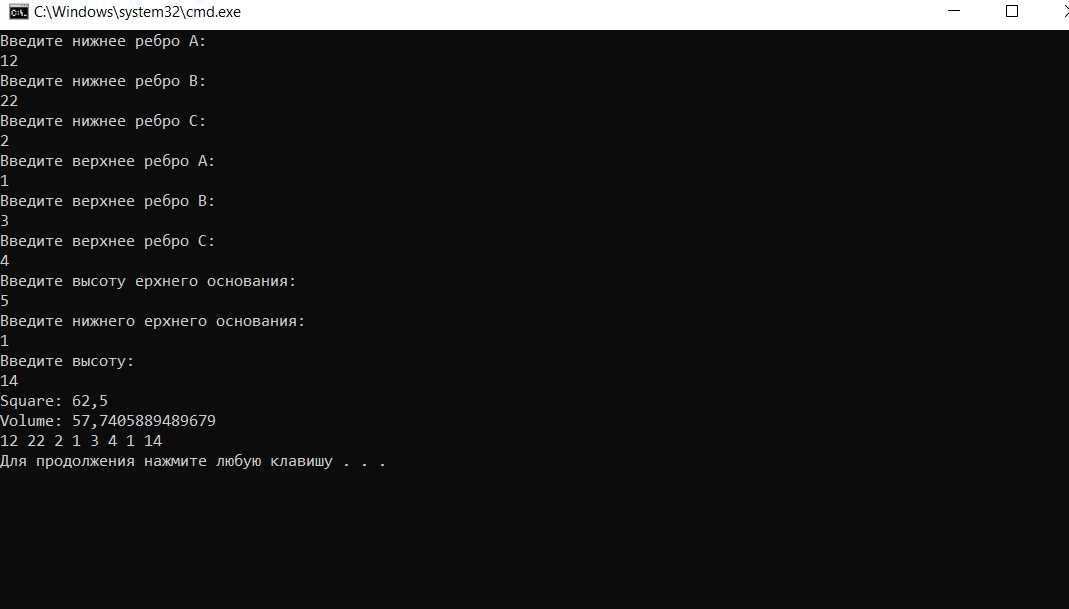
Реализовать множественное наследование.

1. К классам из лабораторной работы №12 часть 2 добавить интерфейс IAdd и ISowInfo, в котором описываются методы ввода-вывода данных согласно параметрам ваших классов, а реализуются уже в нужном классе.
2. К текущему заданию добавить интерфейс ISort, в котором описываются методы сортировки согласно параметрам ваших классов, а реализуются уже в нужном классе.
3. К текущему заданию добавить еще какой-либо свой полезный интерфейс, в котором описываются методы (минимум 3) согласно параметрам ваших классов, а реализуются уже в нужном классе.

Продемонстрировать работу каждого из методов интерфейсов для данных класса.

Псы адин: Не забудьте **схему иерархии** ваших классов и интерфейсов!





using lab13.interfaces;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab13

{

class Piramide : IParams, ISumNum, IShowInfo

{

int rebroNijA;

int rebroNijB;

int rebroNijC;

int rebroVerhA;

int rebroVerhC;

int rebroVerhD;

int hVerhOsn;

int hNijOsn;

int h;

public Piramide(int rebroNijA, int rebroNijB, int rebroNijC, int rebroVerhA, int rebroVerhC, int rebroVerhD, int hVerhOsn, int hNijOsn, int h)

{

this.rebroNijA = rebroNijA;

this.rebroNijB = rebroNijB;

this.rebroNijC = rebroNijC;

this.rebroVerhA = rebroVerhA;

this.rebroVerhC = rebroVerhC;

this.rebroVerhD = rebroVerhD;

this.hVerhOsn = hVerhOsn;

this.hNijOsn = hNijOsn;

this.h = h;

}

public int RebroNijA { get => rebroNijA; set => rebroNijA = value; }

public int RebroNijB { get => rebroNijB; set => rebroNijB = value; }

public int RebroNijC { get => rebroNijC; set => rebroNijC = value; }

public int RebroVerhA { get => rebroVerhA; set => rebroVerhA = value; }

public int RebroVerhC { get => rebroVerhC; set => rebroVerhC = value; }

public int RebroVerhD { get => rebroVerhD; set => rebroVerhD = value; }

public int HVerhOsn { get => hVerhOsn; set => hVerhOsn = value; }

public int HNijOsn { get => hNijOsn; set => hNijOsn = value; }

public int H { get => h; set => h = value; }

public int PerimetrOsn(int rebroA, int rebroB, int rebroC)

{

return rebroA + rebroB + rebroC;

}

public void ShowInfo()

{

Console.WriteLine(RebroNijA + " " + rebroNijB + " " + RebroNijC + " " + RebroVerhA + " " + RebroVerhC + " " + rebroVerhD + " " + HNijOsn + " " + H);

}

public double Square(int hOsn,int rebro)

{

return 1 / 2.0 \* hOsn \* rebro;

}

public int SumNum(int num)

{

int sum = 0;

for (; num!=0; num/=10)

{

sum += num % 10;

}

return sum;

}

public double Volume()

{

return 1 / 3.0 \* h \* (Square(hNijOsn, rebroNijA) + Math.Sqrt(Square(hNijOsn, rebroNijA) \* Square(hVerhOsn, rebroVerhA)) + Square(hVerhOsn, rebroVerhA));

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab13

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Введите нижнее ребро А: ");

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

Console.WriteLine("Введите нижнее ребро B: ");

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

Console.WriteLine("Введите нижнее ребро C: ");

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

Console.WriteLine("Введите верхнее ребро A: ");

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

Console.WriteLine("Введите верхнее ребро B: ");

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

Console.WriteLine("Введите верхнее ребро C: ");

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

Console.WriteLine("Введите высоту ерхнего основания: ");

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

Console.WriteLine("Введите нижнего ерхнего основания: ");

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

Console.WriteLine("Введите высоту: ");

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

Piramide p = new Piramide(rebroNijA, rebroNijB, rebroNijC, rebroVerhA, rebroVerhB, rebroVerhC, hVerhOsn, hNijOsn, h);

Console.WriteLine("Square: " + p.Square(p.HNijOsn, p.RebroNijA) + p.Square(hVerhOsn, rebroVerhA));

Console.WriteLine("Volume: " + p.Volume());

p.ShowInfo();

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab13.interfaces

{

interface IParams

{

double Square(int hOsn, int rebro);

int PerimetrOsn(int rebroA, int rebroB, int rebroC);

double Volume();

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab13.interfaces

{

interface IShowInfo

{

void ShowInfo();

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace lab13.interfaces

{

interface ISumNum

{

int SumNum(int num);

}

}