МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ А.А. КУЛЕШОВА»

СОЦИАЛЬНО-ГУМАНИТАРНЫЙ КОЛЛЕДЖ

ОТЧЁТ

По лабораторной работе № 4

**Разработка программ с использованием операторов цикла**

Учащийся Терещенко С.Г.

Группа 3ПОБШ

Проверил (-а) Дробышевская О.А.

Могилев 2019

**Цель**

− Познакомить с классами в С#.

− Дать описание процессу создания класса.

− Познакомить с атрибутами и методами классов.

− Дать описание методам доступа к элементам класса.

Содержание лабораторной работы

Вариант 12

Реализовать класс (за основу принять класс Rectangle). Количество и содержание методов проработать самостоятельно



using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_5.\_1

{

class Program

{

class ThreePrizm

{

double a { get; set; }

double b { get; set; }

double c { get; set; }

double h { get; set; }

public string classname { get; set; } = "ThreePrizm";

public ThreePrizm()

{

Console.Write("a = "); a = double.Parse(Console.ReadLine());

Console.Write("b = "); b = double.Parse(Console.ReadLine());

Console.Write("c = "); c = double.Parse(Console.ReadLine());

Console.Write("h = "); h = double.Parse(Console.ReadLine());

}

public double BaseSquare()

{

double p = (a + b + c) / 2;

return Math.Sqrt(p \* (p - a) \* (p - b) \* (p - c));

}

public (double a, double b, double c, double h) ReturnParameters()

{

return (a, b, c, h);

}

public double Volume()

{

double p = (a + b + c) / 2;

double s = Math.Sqrt(p \* (p - a) \* (p - b) \* (p - c));

return s \* h;

}

}

static void Main(string[] args)

{

ThreePrizm prizm = new ThreePrizm();

Console.WriteLine("Площадь основания: " + prizm.BaseSquare());

Console.WriteLine("Объем: " + prizm.Volume());

Console.WriteLine("Имя класса: " + prizm.classname);

Console.Write("Введите новое название класса: ");

prizm.classname = Console.ReadLine();

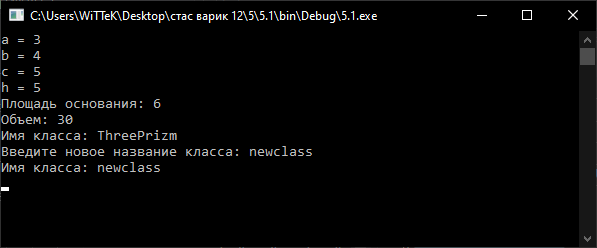
Console.WriteLine("Имя класса: " + prizm.classname);

Console.ReadKey();

}

}

}



Создать класс Trapeze, разработав следующие элементы класса:

а. Поля: int a, b;

б. Конструктор, позволяющий создать экземпляр класса с заданными длинами сторон.

в. Методы, позволяющие:

• вывести длины сторон трапеции на экран;

• расчитать периметр трапеции;

• расчитать площадь трапеции.

г. Свойства:

• получить-установить длины сторон прямоугольника (доступное для чтения и записи);

• позволяющее установить, является ли данная трапеция равнобедренной (доступное только для чтения).

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_5.\_2

{

class Trapeze

{

int a { get; set; }

int b { get; set; }

int c, d;

public Trapeze(int aa, int bb, int cc, int dd)

{

a = aa;//меньшая параллель

b = bb;//большая параллель

c = cc;//боковая 1

d = dd;//боковая 2

}

public (int a, int b, int c, int d) GetLengths()

{

return (a, b, c, d);

}

public int GetPerimeter()

{

return a + b + c + d;

}

public double GetSquare()

{

return (a + b) / 2 \* Math.Sqrt(c \* c - (c \* c - d \* d + (b - a) \* (b - a)) / (2 \* (b - a)));

}

public bool Ravnost()

{

if (c == d) return true;

else return false;

}

public void SetLength()

{

a = int.Parse(Console.ReadLine());

b = int.Parse(Console.ReadLine());

c = int.Parse(Console.ReadLine());

d = int.Parse(Console.ReadLine());

}

}

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Введите сначала две параллели(меньшую, затем большую), а после две боковые стороны");

Trapeze trapeze = new Trapeze(int.Parse(Console.ReadLine()), int.Parse(Console.ReadLine()), int.Parse(Console.ReadLine()), int.Parse(Console.ReadLine()));

Console.WriteLine("Периметр: " + trapeze.GetPerimeter());

Console.WriteLine("Стороны: " + trapeze.GetLengths());

Console.WriteLine("Площадь: " + trapeze.GetSquare());

if (trapeze.Ravnost()) Console.WriteLine("Треугольник равнобедренный");

else Console.WriteLine("Треугольник неравнобедренный");

Console.ReadKey();

}

}

}



**КОНТРОЛЬНЫЕ ВОПРОСЫ**

1 Что понимается под термином «класс»?

Класс – это абстрактное понятие, сравнимое с понятием категория в его обычном смысле. Класс в объектно-ориентированном программировании – это абстрактный тип данных, который включает в себя не только данные, но и функции и процедуры.

2 Какие элементы определяются в составе класса?

Данные, функции и локальные типы.

3 Каково соотношение понятий «класс» и «объект»?

Класс – описание множества объектов и выполняемых над ними действий.

4 Что понимается под термином «члены класса»?

В классах и структурах есть члены, представляющие их данные и поведение. Члены класса включают все члены, объявленные в этом классе, а также все члены (кроме конструкторов и методов завершения), объявленные во всех классах в иерархии наследования данного класса.

5 Перечислите пять разновидностей членов класса специфичных для языка C#.

6 Что понимается под термином «конструктор»?

Конструктор определяется подобно методу за исключением того, что вместо имени метода и возвращаемого типа указывается имя типа, к которому относится этот конструктор.

7 Сколько конструкторов может содержать класс языка C#?

Класс или структура может иметь несколько конструкторов, принимающих различные аргументы.

8 Приведите синтаксис описания класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.

Class <Имя\_класса>

{Секция данных (поля класса)

Секция методов}

9 Какие модификаторы типа доступа Вам известны?

Модификаторы доступа служат для ограничения доступа как к полям класса, так и к методам класса. Ограничения необходимы для того, что бы облегчить создание сложных программных систем: private и public.

10 В чем заключаются особенности доступа членов класса с модификатором public?

Если поле или метод класса определены с модификатором доступа

public, они доступны вне объявления базового класса или производных классов. Это, в частности, означает, что методы, объявленные вне класса, могут свободно обращаться к таким полям и методам.

11 В чем заключаются особенности доступа членов класса с модификатором private?

Модификатор private запрещает доступ к полям и методам класса извне самого класса. Поля и методы, объявленные с модификатором private, будут доступны только в методах самого класса.

12 В чем заключаются особенности доступа членов класса с модификатором protected?

Доступ к типу или члену возможен только из кода в том же классе либо в классе, производном от этого класса.

13 В чем заключаются особенности доступа членов класса с модификатором internal?

Доступ к типу или члену возможен из любого кода в той же сборке, но не из другой сборки.

14 Какое ключевое слово языка C# используется при создании объекта?

Объекты можно создавать с помощью ключевого слова new, за которым следует имя класса, на котором будет основан объект

15 Приведите синтаксис создания объекта в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.

Customer object1 = new Customer();

16 В чем состоит назначение конструктора?

Конструкторы выполняют код инициализации класса или структуры. Конструкторы позволяют программисту задавать значения по умолчанию, ограничивать число установок и писать код, который является гибким и удобным для чтения.

17 Каждый ли класс языка C# имеет конструктор?

Да

18 Какие умолчания для конструкторов приняты в языке C#?

Каждый раз, когда создается класс или структура, вызывается конструктор. Конструкторы имеют имя, совпадающее с именем класса или структуры, и обычно инициализируют члены данных нового объекта. Конструктор, который не принимает никаких параметров, называется конструктором без параметров. Конструкторы без параметров вызываются всякий раз, когда создается экземпляр объекта с помощью оператора new, а аргументы в new не передаются.

19 Каким значением инициализируются по умолчанию значения ссылочного типа?

Null

20 В каком случае конструктор по умолчанию не используется?

Если класс абстрактный.

21 Приведите синтаксис конструктора класса в общем виде. Проиллюстрируйте его фрагментом программы на языке C#.

