УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

**“ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО”**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 1

по дисциплине «Объектно-ориентированное программирование»

На тему **«**Основы синтаксиса объектно-ориентированного языка программирования**»**

Выполнил:

студент гр. ИТП-21

Мурашко М.Д.

Принял: ассистент

Гуменников Е.Д.

Гомель 2022

**Цель работы:** Освоить основы синтаксиса объектно-ориентированного языка программирования на языке *С#*.

**Задание:**

1. Необходимо разработать класс, представляющий собой плоскую фигуру, согласно варианта (таблица 1).
2. При именовании класса, полей и методов руководствоваться соглашением о наименовании кода
3. Фигуру следует задавать координатами вершин.
4. Реализовать метод проверки возможности существования данной фигуры
5. Реализовать методы вычисление длин сторон, площади и периметра
6. Реализовать методы, проверяющие принадлежность точки, заданной своими координатами на плоскости, фигуре, её границе.
7. Класс должен быть размещён в библиотеке классов.
8. Весь код должен быть снабжён элементами документирования
9. Сгенерировать XML-файл с документацией по проекту
10. Создать консольное приложение с интерфейсом пользователя
11. Подключить созданную библиотеку в консольное приложение
12. Верифицировать разработанное приложение
13. Сгенерировать исполняемый файл для последующего переноса на другие компьютеры

Таблица 1

|  |  |
| --- | --- |
| **Вариант** | **Условие** |
| 3 | Трапеция |

**Ход работы**

В ходе работы был разработан класс *Trapeze,* который представляет плоскую фигуру, заданную координатами четырех точек. В классе трапеции хранятся такие переменные какмассивы координат точек по осям X и Y. Так же класс содержит методы для: выяснения возможности существования данной трапеции, вычисления: площади, длины, периметра и проверки точки на принадлежность к фигуре. Код основной программы представлен в приложении А, а код библиотеки классов в приложении В.

Результат работы представлен на рисунке 1:

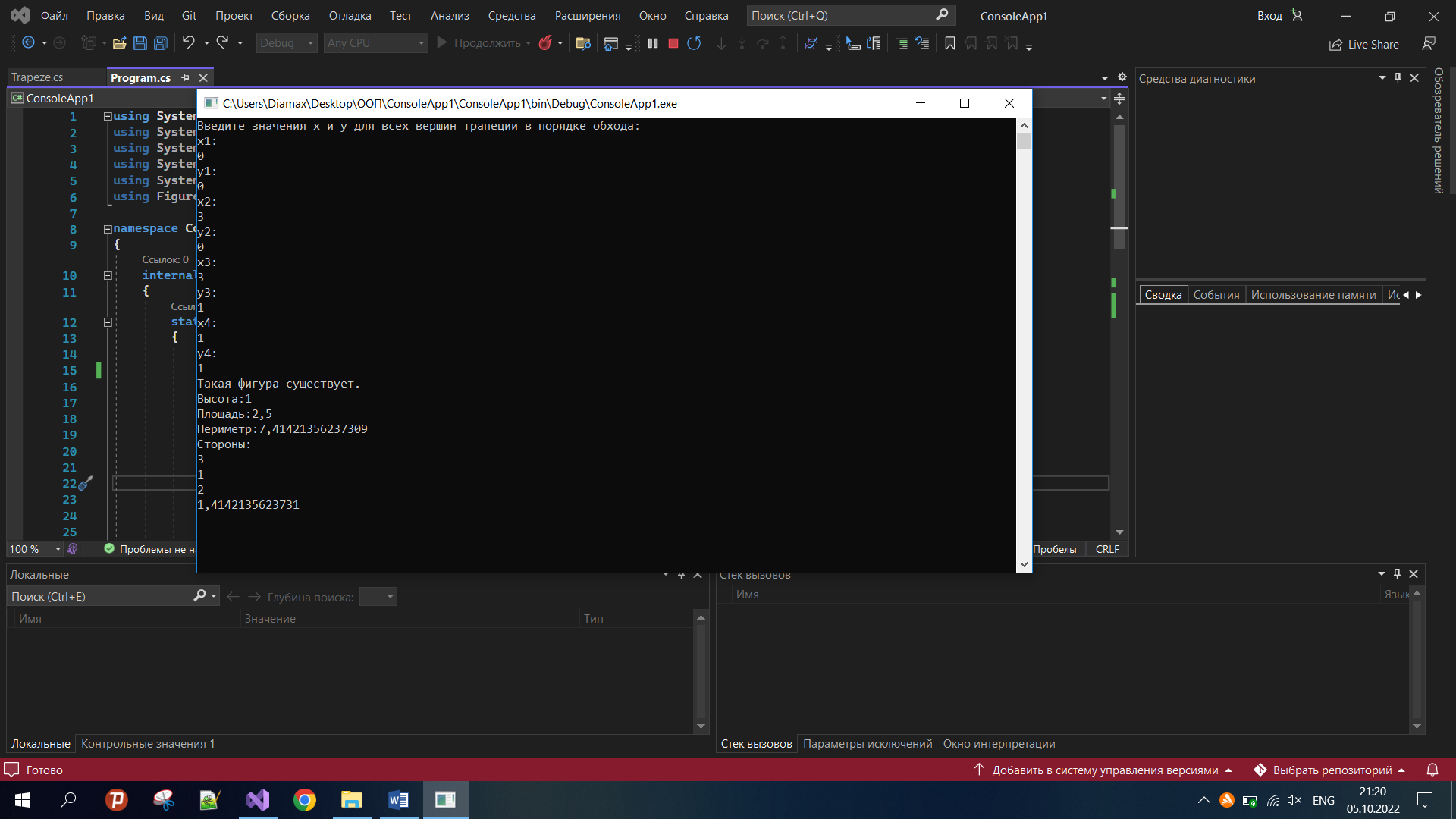


Рисунок 1 – Выполнение программы в консоли

**Вывод:** в ходе лабораторной работы были освоены основы синтаксиса объектно-ориентированного языка программирования на языке *c#*.

**Приложение А**

Листинг основной программы

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Figure;

namespace ConsoleApp1

{

internal class Program

{

static void Main(string[] args)

{

double x1, x2, x3, x4, y1, y2, y3, y4;

Console.WriteLine("Введите значения x и y для всех вершин трапеции в порядке обхода:");

Console.WriteLine("x1:");

x1 = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("y1:");

y1 = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("x2:");

x2 = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("y2:");

y2 = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("x3:");

x3 = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("y3:");

y3 = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("x4:");

x4 = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("y4:");

y4 = Convert.ToDouble(Console.ReadLine());

Figure.Trapezoid t = new Figure.Trapezoid(x1, x2, x3, x4, y1, y2, y3, y4);

double[] sides = t.GetSides();

if (t.IsReal())

{

Console.WriteLine("Такая фигура существует.");

Console.WriteLine("Высота:" + t.h);

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

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

Console.WriteLine("Cтороны:");

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

{

Console.WriteLine(sides[i]);

}

}

else

{

Console.WriteLine("Такая фигура не может существовать!");

}

Console.ReadKey();

}

}

}

**Приложение В**

Листинг библиотеки классов Trapezoid

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Figure

{

public class Trapezoid

{

public double x1, x2, x3, x4, y1, y2, y3, y4,Side1,Side2,Side3,Side4,

h;//Высота трапеции

bool flag;//переменная для определения какие стороны-основания,а какие боковые;

public Trapezoid(double x1, double x2, double x3, double x4, double y1, double y2, double y3, double y4)

{

this.x1 = x1;

this.x2 = x2;

this.x3 = x3;

this.x4 = x4;

this.y1 = y1;

this.y2 = y2;

this.y3 = y3;

this.y4 = y4;

}

public bool IsReal()

{

//Четырёхугольник-трапеция,если две стороны параллельны,а две другие нет

//Два вектора параллельны,если cos угла между ними равен 1

double cos1=(Math.Abs((x2 - x1)) \* Math.Abs((x3 - x4)) + Math.Abs(y1 - y2) \* Math.Abs(y3 - y4)) / (Math.Sqrt(Math.Pow(Math.Abs(x2-x1),2) + Math.Pow(Math.Abs(y2 - y1), 2)) \* Math.Sqrt(Math.Pow(Math.Abs(x3 - x4), 2) + Math.Pow(Math.Abs(y3 - y4), 2)));

double cos2 =(Math.Abs((x4 - x1)) \* Math.Abs((x3 - x2)) + Math.Abs(y4 - y1) \* Math.Abs(y3 - y2)) / (Math.Sqrt(Math.Pow(Math.Abs(x4 - x1), 2) + Math.Pow(Math.Abs(y4 - y1), 2)) \* Math.Sqrt(Math.Pow(Math.Abs(x3 - x2), 2) + Math.Pow(Math.Abs(y3 - y2), 2)));

if (cos1 == 1 && cos2 != 1)

{

h = Math.Sqrt(Math.Pow(x4 - x1, 2) + Math.Pow(y4 - y1, 2) - Math.Pow((Math.Abs(x4-x1)),2));

flag = true;

return true;

}

else if (cos1 != 1 && cos2 == 1)

{

h = Math.Sqrt(Math.Pow(x3 - x4, 2) + Math.Pow(y3 - y4, 2) - Math.Pow((Math.Abs(x3 - x4)), 2));

return true;

}

else

{

return false;

}

}

public double[] GetSides()

{

Side1 = (double)(Math.Sqrt(Math.Pow(x1-x2,2)+ Math.Pow(y1 - y2,2)));

Side2 = (double)(Math.Sqrt(Math.Pow(x2 - x3,2)+ Math.Pow(y2 - y3, 2)));

Side3 = (double)(Math.Sqrt(Math.Pow(x3 - x4,2)+ Math.Pow(y3 - y4, 2)));

Side4 = (double)(Math.Sqrt(Math.Pow(x4 - x1,2)+ Math.Pow(y4 - y1, 2)));

double[] sides = {Side1,Side2,Side3,Side4};

return sides;

}

public double GetPerimeter()

{

return Side1 + Side2 + Side3 + Side4;

}

public double GetSquare()

{

if (flag)

{

return (0.5 \* h \* (Side1+Side3));//полусумма оснований на высоту

}

else {

return (0.5 \* h \* (Side4 + Side2));

}

}

}

}