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

Трацевский И.С.

Принял: ассистент

Гуменников Е.Д.

Гомель 2022

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

**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 Задачи** |
| 7 | Параллелограмм |

**Ход работы**

В ходе работы был разработан класс *Parallel,* который представляет плоскую фигуру, заданную координатами четырех точек. В классе параллелограмма хранятсямассивы координат точек по осям X и Y. Так же класс содержит методы для: выяснения возможности существования данного параллелограмма, вычисления: площади, длины, периметра – и проверки точки на принадлежность к фигуре. Код программы будет представлен в приложении А.

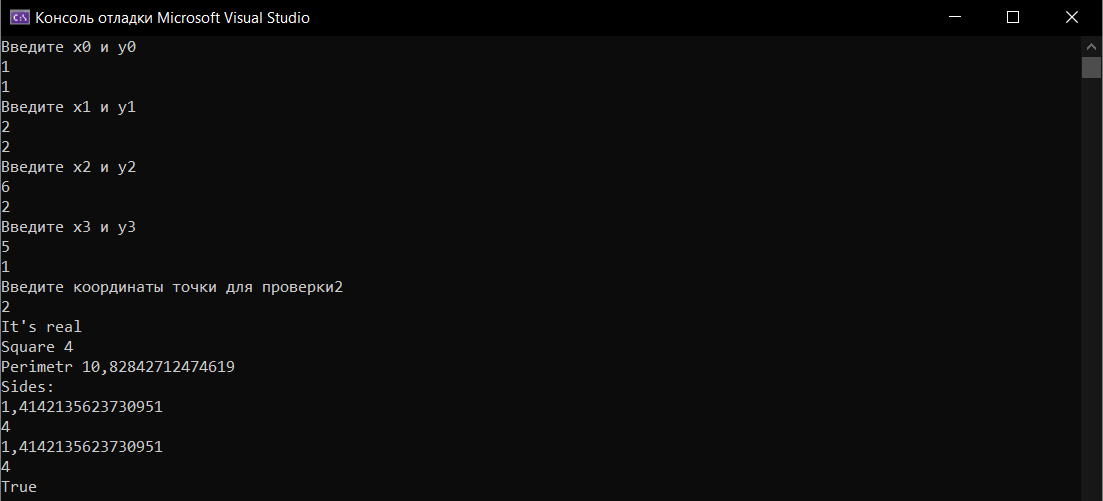


Рисунок 1 – Пример работы программы

**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 lab1norm

{

internal class Program

{

static void Main(string[] args)

{

double[] x = new double[4], y = new double[4];

double pointX,pointY;

for(int i = 0; i < 4; i++)

{

Console.Write($"Введите x{i} и y{i} \n");

x[i] = Convert.ToDouble(Console.ReadLine());

y[i] = Convert.ToDouble(Console.ReadLine());

}

Console.Write("Введите координаты точки для проверки");

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

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

Figure.Parallel p1 = new Figure.Parallel(x,y);

double[] sides = p1.getSides();

if (p1.isReal())

{

Console.WriteLine("It's real");

Console.WriteLine("Square " + p1.getSquare());

Console.WriteLine("Perimetr " + p1.getPerimeter());

Console.WriteLine("Sides:");

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

{

Console.WriteLine(sides[i]);

}

Console.WriteLine(p1.doesInclude(0, 0));

}

else

{

Console.WriteLine("It's not real");

}

}

}

}

Листинг класса *Parallel*

namespace Figure

{

internal class Parallel

{

public double side1, side2, side3, side4;

public double[] x = new double[4], y = new double[4];

public Parallel(double[] x, double[] y)

{

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

{

this.x[i] = x[i];

this.y[i] = y[i];

}

}

public bool isReal()

{

return side1 == side3 && side2 == side4;

}

private void setSides()

{

side1 = (double)(Math.Sqrt(Math.Pow(x[0] - x[1], 2) + Math.Pow(y[0] - y[1], 2)));

side2 = (double)(Math.Sqrt(Math.Pow(x[1] - x[2], 2) + Math.Pow(y[1] - y[2], 2)));

side3 = (double)(Math.Sqrt(Math.Pow(x[2] - x[3], 2) + Math.Pow(y[2] - y[3], 2)));

side4 = (double)(Math.Sqrt(Math.Pow(x[3] - x[0], 2) + Math.Pow(y[3] - y[0], 2)));

}

public double[] getSides()

{

setSides();

double[] sides = { side1, side2, side3, side4 };

return sides;

}

public double getPerimeter()

{

return side1 + side2 + side3 + side4;

}

public double getSquare()

{

double heightLength = x[1] - x[0];

return Math.Abs(heightLength \* side4);

}

public bool doesInclude(double pointX, double pointY)

{

bool includeS1 = (pointX - x[0]) \* (y[1] - y[0]) == (pointY - y[0]) \* (x[1] - x[0]);

bool includeS2 = (pointX - x[1]) \* (y[2] - y[1]) == (pointY - y[1]) \* (x[2] - x[1]);

bool includeS3 = (pointX - x[2]) \* (y[3] - y[2]) == (pointY - y[2]) \* (x[3] - x[2]);

bool includeS4 = (pointX - x[3]) \* (y[0] - y[3]) == (pointY - y[3]) \* (x[0] - x[3]);

return includeS1 || includeS2 || includeS3 || includeS4;

}

}

}

Начало формы

Конец формы