**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ**

**УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ЛАБОРАТОРНАЯ РАБОТА №1

По дисциплине: **«Объектно-ориентированное программирование»**

на тему: **Основы синтаксиса объектно-ориентированного языка программирования**

Выполнил: студент группы ИТП-21

Пронуза М.Ю.

Проверил:

Малиновский И.Л.

Гомель 2023

**Цель работы**: изучить основы синтаксиса объектно-ориентированного языка программирования. Научиться разрабатывать классы, соблюдая принципы объектно-ориентированного программирования. Определять различные методы для заданной фигуры.

**Задание:**

1. Необходимо разработать класс, представляющий собой плоскую фигуру, согласно варианту (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.

**Ход выполнения и результаты работы:**

Текстовый вариант задания для варианта 11 представлен на рисунке 1.



Рисунок 1 – Задание варианта 4

На рисунке 2 представлен результат выполнения программы.

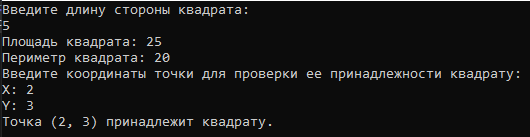


Рисунок 2 – Результат выполнения программы

На рисунке 4 представлен сгенерированный XML-файл с документацией по проекту.

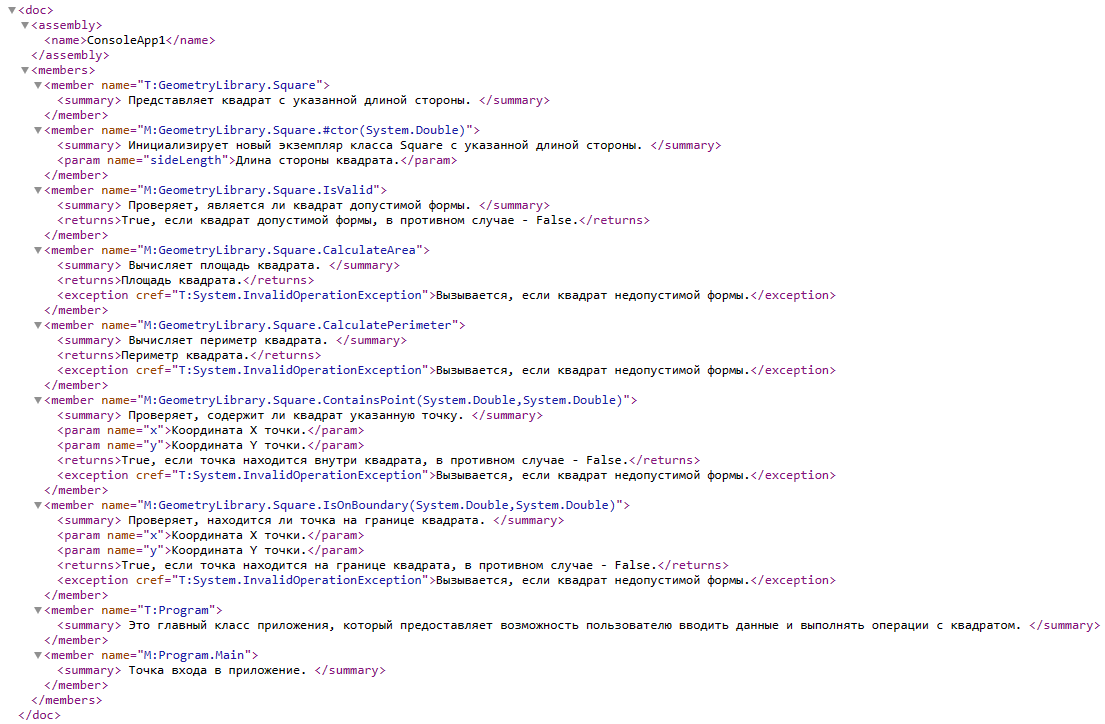


Рисунок 4 – XML-файл с документацией

Текст класса *GeometryLibrary* представлен в приложении А.

Текст класса *Program* представлен в приложении Б.

**Вывод:** в ходе выполнения лабораторной работы изучены основы синтаксиса объектно-ориентированного языка программирования, использование различных классов. В результате выполнения лабораторной работы разработана программа для создания фигуры и расчета её данных.

**ПРИЛОЖЕНИЕ А**

**Текст GeometryLibrary.cs**

using System;

namespace GeometryLibrary

{

/// <summary>

/// Представляет квадрат с указанной длиной стороны.

/// </summary>

public class Square

{

private double sideLength;

private double x;

private double y;

/// <summary>

/// Инициализирует новый экземпляр класса Square с указанной длиной стороны.

/// </summary>

/// <param name="sideLength">Длина стороны квадрата.</param>

public Square(double sideLength, double x, double y)

{

this.sideLength = sideLength;

this.x = x;

this.y = y;

if (!IsValid())

{

throw new InvalidOperationException("Квадрат недопустимой формы");

}

}

/// <summary>

/// Проверяет, является ли квадрат допустимой формы.

/// </summary>

/// <returns>True, если квадрат допустимой формы, в противном случае - False.</returns>

public bool IsValid()

{

return sideLength > 0;

}

/// <summary>

/// Вычисляет площадь квадрата.

/// </summary>

/// <returns>Площадь квадрата.</returns>

/// <exception cref="InvalidOperationException">Вызывается, если квадрат недопустимой формы.</exception>

public double CalculateArea()

{

return sideLength \* sideLength;

}

/// <summary>

/// Вычисляет периметр квадрата.

/// </summary>

/// <returns>Периметр квадрата.</returns>

/// <exception cref="InvalidOperationException">Вызывается, если квадрат недопустимой формы.</exception>

public double CalculatePerimeter()

{

return 4 \* sideLength;

}

/// <summary>

/// Проверяет, содержит ли квадрат указанную точку.

/// </summary>

/// <param name="x">Координата X точки.</param>

/// <param name="y">Координата Y точки.</param>

/// <returns>True, если точка находится внутри квадрата, в противном случае - False.</returns>

/// <exception cref="InvalidOperationException">Вызывается, если квадрат недопустимой формы.</exception>

public bool ContainsPoint(double x, double y)

{

return x-this.x > 0 && x-this.x < sideLength && y-this.y > 0 && y-this.y < sideLength;

}

/// <summary>

/// Проверяет, находится ли точка на границе квадрата.

/// </summary>

/// <param name="x">Координата X точки.</param>

/// <param name="y">Координата Y точки.</param>

/// <returns>True, если точка находится на границе квадрата, в противном случае - False.</returns>

/// <exception cref="InvalidOperationException">Вызывается, если квадрат недопустимой формы.</exception>

public bool IsOnBoundary(double x, double y)

{

return (x-this.x == 0 || x - this.x == sideLength) && (y - this.y >= 0 && y - this.y <= sideLength) ||

(y - this.y == 0 || y - this.y == sideLength) && (x - this.x >= 0 && x - this.x <= sideLength);

}

}

}

**ПРИЛОЖЕНИЕ Б**

**Текст Program.cs**

using System;

using GeometryLibrary;

/// <summary>

/// Это главный класс приложения, который предоставляет возможность пользователю вводить данные

/// и выполнять операции с квадратом.

/// </summary>

class Program

{

/// <summary>

/// Точка входа в приложение.

/// </summary>

static void Main()

{

Console.WriteLine("Введите длину стороны квадрата:");

double sideLength = double.Parse(Console.ReadLine());

Console.WriteLine("Введите x:");

double sx = double.Parse(Console.ReadLine());

Console.WriteLine("Введите y:");

double sy= double.Parse(Console.ReadLine());

// Создаем экземпляр квадрата на основе введенной длины стороны.

Square square = new Square(sideLength, sx, sy);

if (square.IsValid())

{

Console.WriteLine($"Площадь квадрата: {square.CalculateArea()}");

Console.WriteLine($"Периметр квадрата: {square.CalculatePerimeter()}");

Console.WriteLine("Введите координаты точки для проверки ее принадлежности квадрату:");

Console.Write("X: ");

double x = double.Parse(Console.ReadLine());

Console.Write("Y: ");

double y = double.Parse(Console.ReadLine());

if (square.ContainsPoint(x, y))

{

Console.WriteLine($"Точка ({x}, {y}) принадлежит квадрату.");

}

else if (square.IsOnBoundary(x, y))

{

Console.WriteLine($"Точка ({x}, {y}) находится на границе квадрата.");

}

else

{

Console.WriteLine($"Точка ({x}, {y}) не принадлежит квадрату.");

}

}

else

{

Console.WriteLine("Недопустимая длина стороны квадрата.");

}

}

}