namespace UKPO\_6

{

public class Point

{

private double \_x;

private double \_y;

public double X

{

get { return \_x; }

set { \_x = value; }

}

public double Y

{

get { return \_y; }

set { \_y = value; }

}

public Point(double x, double y)

{

\_x = x;

\_y = y;

}

}

}

using System;

namespace UKPO\_6

{

public class Program

{

static void Main(string[] args)

{

Console.Write("Введите координату X верхнего левого угла прямоугольника : ");

double x1 = double.Parse(Console.ReadLine());

Console.Write("Введите координату Y верхнего левого угла прямоугольника : ");

double y1 = double.Parse(Console.ReadLine());

Console.Write("Введите координату X нижнего правого угла прямоугольника : ");

double x2 = double.Parse(Console.ReadLine());

Console.Write("Введите координату Y нижнего правого угла прямоугольника : ");

double y2 = double.Parse(Console.ReadLine());

try

{

Rectangle rectangle = new Rectangle(new Point(x1, y1), new Point(x2, y2));

Console.WriteLine($"Периметр : {rectangle.Perimeter()}");

Console.WriteLine($"Площадь : {rectangle.Square()}");

}

catch (Exception ex)

{

Console.WriteLine(ex.Message);

}

}

}

}

using System;

namespace UKPO\_6

{

public class Rectangle

{

private Point \_leftUpper;

private Point \_rightLower;

public Point LeftUpper

{

get { return \_leftUpper; }

set

{

if(\_rightLower != null)

{

if (value.X > \_rightLower.X)

throw new ArgumentException("Левый угол не может быть правее правого", nameof(LeftUpper));

if (value.Y < \_rightLower.Y)

throw new ArgumentException("Верхний угол не может быть ниже нижнего", nameof(LeftUpper));

}

\_leftUpper = value;

}

}

public Point RightLower

{

get { return \_rightLower; }

set

{

if (\_leftUpper != null)

{

if (value.X < \_leftUpper.X)

throw new ArgumentException("Правый угол не может быть левее левого", nameof(RightLower));

if (value.Y > \_leftUpper.Y)

throw new ArgumentException("Нижний угол не может быть выше верхнего", nameof(RightLower));

}

\_rightLower = value;

}

}

public Rectangle(Point leftUpper, Point rightLower)

{

LeftUpper = leftUpper;

RightLower = rightLower;

}

public double Heigth()

{

return \_leftUpper.Y - \_rightLower.Y;

}

public double Width()

{

return \_rightLower.X - \_leftUpper.X;

}

public double Perimeter()

{

return Width() + Heigth() \* 2;

}

public double Square()

{

return Width() \* Heigth();

}

}

}

## Оценка характеристик программы

### Метрика взвешенных методов

### Кол-во методов на класс

### Связность между классами

*Point:* нет обращений к другим классам

*Rectangle:*

* 6 обращений к классу *Point*
* 4 обращения к конструктору класса *ArgumentException*

*Program:*

* 11 обращений к классу *Console*
* 4 обращения к методы Parse класса *double*
* 2 обращения к классу *Point*
* 2 обращения к классу *Rectangle*
* 1 обращение к классу *Exception*

### Кол-во откликов на класс