**17 СБОРКИ, БИБЛИОТЕКИ, АТРИБУТЫ, ДИРЕКТИВЫ**

Задание 1. Треугольник (методы ввода сторон, проверки на существование, вычисления периметра, вычисления площади, определения вида треугольника (разносторонний, равнобедренный, равносторонний)); Прямоугольник (методы ввода сторон, вычисления периметра, вычисления площади).

Листинг программы:

namespace FigureLib

{

public abstract class Figure

{

public abstract string Area();

public abstract string Perimeter();

public abstract string ShapeName();

public abstract string GetFigureType();

}

public class Triangle : Figure

{

double sideA, sideB, sideC; // Стороны треугольника

public Triangle(double triangleSideA, double triangleSideB, double triangleSideC)

{

if (!isExist(triangleSideA, triangleSideB, triangleSideC))

{

throw new Exception($"Error");

}

SideA = triangleSideA;

SideB = triangleSideB;

SideC = triangleSideC;

}

public double SideA

{

get { return sideA; }

set { sideA = value < 0 ? -value : value; }

}

public double SideB

{

get { return sideB; }

set { sideB = value < 0 ? -value : value; }

}

public double SideC

{

get { return sideC; }

set { sideC = value < 0 ? -value : value; }

}

public bool isExist(double sideA, double sideB, double sideC)

{

if (sideA > 0 && sideB > 0 && sideC > 0 && (sideA + sideB > sideC && sideA + sideC > sideB && sideB + sideC > sideA))

{

return true;

}

return false;

}

public override string Area()

{

double semPer = (sideA + sideB + sideC) / 2;

return Math.Sqrt(semPer \* (semPer - sideA) \* (semPer - sideB) \* (semPer - sideC)).ToString();

}

public override string Perimeter()

{

return (sideA + sideB + sideC).ToString();

}

public override string GetFigureType()

{

if (sideA == SideB && sideA == sideC) return "равносторонний";

if (sideA == SideB || sideA == sideC || SideB == sideC) return "равнобедренный";

return "треугольник, как треугольник";

}

public override string ShapeName()

{

return "Треугольник";

}

}

}

namespace ClassLibrary1

{

public abstract class RectangleFigure

{

public abstract string Area();

public abstract string Perimeter();

public abstract string ShapeName();

public abstract string GetFigureType();

}

public class Rectangle : RectangleFigure

{

double width; // Ширина прямоугольника

double height; // Высота прямоугольника

public Rectangle(double rectangleWidth, double rectangleHeight)

{

Width = rectangleWidth;

Height = rectangleHeight;

}

public double Width

{

get { return width; }

set { width = value < 0 ? -value : value; }

}

public double Height

{

get { return height; }

set { height = value < 0 ? -value : value; }

}

public override string Area()

{

return (width \* height).ToString();

}

public override string Perimeter()

{

return (width \* 2 + height \* 2).ToString();

}

public override string GetFigureType()

{

if (width == height) return "Квадрат";

return "Не квадратный прямоугольник";

}

public override string ShapeName()

{

return "Прямоугольник";

}

}

}