**17.** **Сборки, библиотеки, атрибуты, директивы**

Задание №1

Листинг программы:

z1.Program:

using Z1Lib;

Console.WriteLine("Введите сторона прямоугольника:");

Console.Write(" A - ");

double aR = Convert.ToDouble(Console.ReadLine());

Console.Write(" B - ");

double bR = Convert.ToDouble(Console.ReadLine());

Rectangle r1 = new Rectangle(aR, bR);

Console.WriteLine(r1.ToString());

Console.WriteLine("-------------------------------");

Console.WriteLine("Введите сторона треугольника:");

Console.Write(" A - ");

double aT = Convert.ToDouble(Console.ReadLine());

Console.Write(" B - ");

double bT = Convert.ToDouble(Console.ReadLine());

Console.Write(" C - ");

double cT = Convert.ToDouble(Console.ReadLine());

Triangle t1 = new Triangle(aT, bT, cT);

Console.WriteLine(t1.ToString());

Z1Lib.Figure:

namespace Z1Lib

{

public abstract class Figure

{

public abstract double CalculatePerimeter();

public abstract double CalculateSquare();

public abstract override string ToString();

}

}

Z1Lib.Rectangle:

namespace Z1Lib

{

public class Rectangle : Figure

{

private double \_a;

private double \_b;

public double A { get => \_a; }

public double B { get => \_b; }

public Rectangle(double a, double b)

{

\_a = a;

\_b = b;

}

public override double CalculatePerimeter() => \_a \* 2 + \_b \* 2;

public override double CalculateSquare() => \_a \* \_b;

public override string ToString()

{

return $"Стороны:\n" +

$" A:{\_a}\n" +

$" B:{\_b}\n" +

$"Периметр: {CalculatePerimeter()}\n" +

$"Площадь: {CalculateSquare():.###}";

}

}

}

Z1Lib.Triangle:

namespace Z1Lib

{

public class Triangle : Figure

{

private double \_a;

private double \_b;

private double \_c;

public double A { get => \_a; }

public double B { get => \_b; }

public double C { get => \_c; }

public TriangleType Type { get; }

public Triangle(double a, double b, double c)

{

if (CheckingForExistence(a, b, c))

{

\_a = a;

\_b = b;

\_c = c;

Type = DetermineType();

}

else

{

throw new Exception("Данный треугольник не модежет существовать.\n" +

"Треугольник существует только тогда, когда сумма двух его сторон больше третьей.");

}

}

public override double CalculatePerimeter() => \_a + \_b + \_c;

public override double CalculateSquare()

{

double p = CalculatePerimeter() / 2;

return Math.Sqrt(p \* (p - \_a) \* (p - \_b) \* (p - \_c));

}

public override string ToString()

{

return $"Тип треугольника: \n" +

$"Стороны\n" +

$" А:{\_a}\n" +

$" B:{\_b}\n" +

$" C:{\_c}\n" +

$"Периметр: {CalculatePerimeter()}\n" +

$"Площадь: {CalculateSquare():.###}";

}

private bool CheckingForExistence(double a, double b, double c) => a + b > c && b + c > a && a + c > b;

private TriangleType DetermineType()

{

if (\_a != \_b && \_a != \_c && \_b != \_c)

{

return TriangleType.Versatile;

}

else if (\_a == \_b && \_a == \_c && \_b == \_c)

{

return TriangleType.Equilateral;

}

else

{

return TriangleType.Isosceles;

}

}

}

}

Z1Lib.TriangleType:

namespace Z1Lib

{

public enum TriangleType

{

Versatile,

Isosceles,

Equilateral

}

}

Таблица 17.1 – Входные и выходные данные программы задание №1

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Стороны прямоугольника:  A - 4  B – 5  Стороны треугольника:  A - 8  B - 4  C - 9 | Периметр: 18  Площадь: 20  Периметр: 21  Площадь: 15,998 |

Анализ результатов:

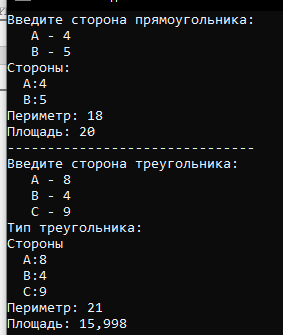


Рисунок 17.1 – Результат выполнения программы задание №1

Задание №2

Листинг программы:

z2.Program:

using Z2Lib.Model;

using Z2Lib.Enum;

Console.WriteLine("Заполните данные: ");

string name = Print("Название: ");

int year = Convert.ToInt32(Print("Год выпуска: "));

GenereType genere = (GenereType)Enum.Parse(typeof(GenereType), Print("Введите жанр: "));

string nameCompany = Print("Название компании(издатель): ");

int numOfCopiesSold = Convert.ToInt32(Print("Кол-во продоных копий: "));

decimal cost = Convert.ToDecimal(Print("Стоимость: "));

Game game = new Game(name, year, genere, new Company(nameCompany),

numOfCopiesSold, cost);

Console.WriteLine();

Console.WriteLine(game);

string Print(string mes = "")

{

Console.Write(mes);

return Console.ReadLine();

}

Z2Lib.Game:

using Z2Lib.Enum;

namespace Z2Lib.Model

{

public class Game

{

private string \_name;

private int \_releaseYearDate;

private GenereType \_genre;

private Company \_creatorCompany;

private int \_numOfCopiesSold;

private decimal \_cost;

public Game(string name, int releaseYearDate, GenereType genre,

Company creatorCompany, int numOfCopiesSold, decimal cost)

{

\_name = name;

if (!(9999 >= releaseYearDate && releaseYearDate >= 1800))

{

throw new Exception("The year is not in the range from 1800 to 9999.\n" +

$"Year = {releaseYearDate}");

}

\_releaseYearDate = releaseYearDate;

\_genre = genre;

\_creatorCompany = creatorCompany;

if (!(numOfCopiesSold >= 0))

{

throw new Exception("The number of copies sold cannot be less than zero.");

}

\_numOfCopiesSold = numOfCopiesSold;

if (!(cost >= 0))

{

throw new Exception("The cost of a copy cannot be less than zero.");

}

\_cost = cost;

}

public override string ToString()

{

return $"Name: {\_name}\n" +

$"Date release: {\_releaseYearDate}\n" +

$"Genre {\_genre}\n" +

$"Creator: {\_creatorCompany}\n" +

$"Number of copies sold: {\_numOfCopiesSold}\n" +

$"Cost: {\_cost}";

}

}

}

Z2Lib.Company:

namespace Z2Lib.Model

{

public class Company

{

private string \_name;

public string Name { get => \_name; }

public Company(string name)

{

\_name = name;

}

public override string ToString()

{

return \_name;

}

}

}

Z2Lib.GenereType:

namespace Z2Lib.Enum

{

public enum GenereType

{

MOBA,

MMORPG,

BattleRoyale

}

}

Таблица 17.2 – Входные и выходные данные программы задание №2

|  |  |
| --- | --- |
| Входные данные | Выходные данные |
| Название: НазваниеИгры  Год выпуска: 2000  Введите жанр: MOBA  Название компании(издатель): CompanyName  Кол-во продоных копий: 1243  Стоимость: 12 | Name: НазваниеИгры  Date release: 2000  Genre MOBA  Creator: CompanyName  Number of copies sold: 1243  Cost: 12 |

Анализ результатов:

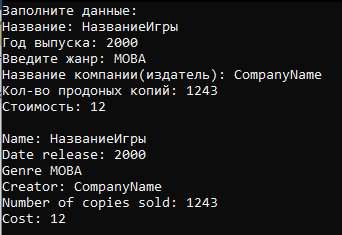


Рисунок 17.2 – Результат выполнения программы задание №2