Лабораторная работа №2

«Реализация метода пошаговой детализации на основе использования объектно-ориентированного программирования (ООП)»

**Задание**

Разработать программу для расчёта всех углов треугольника на основе входных данных.

**Определить все углы треугольника** при заданных значениях сторон **a, b, c.**

**Назначение метода:** Вычисление угла по теореме косинусов:

Для решения задачи следует создать следующий интерфейс, а также несколько кнопок. В первой из них должен содержаться код, в котором создаётся конструктор без параметров, поля для ввода данных в объект **TextBox**, выполнена проверка на корректность ввода данных, отвечающая за вывод соответствующего сообщения в случае, если пользователь ввёл стороны треугольника, любая из сумм двух сторон которого меньше третей.

Для реализации корректности данных написать метод **Correct**, далее осуществить вызов следующих методов:

public void **Correct**() – метод проверки корректности введённых данных;

public double **CalculateAngle**(double a, double b, double c) – метод для расчёта одного угла по теореме косинусов;

private double **RadiansToDegree**(double rad) – метод для преобразования радиан в градусы.

При обращении к ним должны использоваться закрытые поля **\_sideA, \_sideB, \_sideC, \_angleA, \_angleB, \_angleC,** доступ к которым будет только по соответствующим свойствам.

Во второй кнопке экземпляр класса должен создаваться через конструктор с параметрами, поля для ввода остаются теми же, проверка на корректность ввода данных должна присутствовать внутри самого конструктора.

* 1. **Разработка программного кода**

1. using MainDLL;
2. using System;
3. using System.Windows.Forms;
4. namespace MainApplication
5. {
6. public partial class ClassesDemo : Form
7. {
8. public ClassesDemo()
9. {
10. InitializeComponent();
11. }
12. public class Triangle
13. {
14. public bool IsOK
15. { get; set; }
16. private double \_sideA;
17. public double SideA
18. {
19. get => \_sideA;
20. set
21. {
22. if (value > 0)
23. \_sideA = value;
24. else
25. IsOK = false;
26. }
27. }
28. private double \_sideB;
29. public double SideB
30. {
31. get => \_sideB;
32. set
33. {
34. if (value > 0)
35. \_sideB = value;
36. else
37. IsOK = false;
38. }
39. }
40. private double \_sideC;
41. public double SideC
42. {
43. get => \_sideC;
44. set
45. {
46. if (value > 0)
47. \_sideC = value;
48. else
49. IsOK = false;
50. }
51. }
52. private double \_angleA;
53. public double AngleA
54. {
55. get => \_angleA;
56. set { \_angleA = value; }
57. }
58. private double \_angleB;
59. public double AngleB
60. {
61. get => \_angleB;
62. set { \_angleB = value; }
63. }
64. private double \_angleC;
65. public double AngleC
66. {
67. get => \_angleC;
68. set { \_angleC = value; }
69. }
70. public Triangle() // конструктор без параметров
71. {
72. }
73. public Triangle(double sideA, double sideB, double sideC) // конструктор с параметрами
74. {
75. if (SideA + SideB < SideC || SideA + SideC < SideB || SideB + SideC < SideA)
76. {
77. this.IsOK = false;
78. }
79. else
80. {
81. this.IsOK = true;
82. this.SideA = sideA;
83. this.SideB = sideB;
84. this.SideC = sideC;
85. }
86. this.Correct();
87. }
88. // метод для проверки реальности предполагаемого треугольника
89. public void Correct()
90. {
91. if (SideA + SideB < SideC || SideA + SideC < SideB || SideB + SideC < SideA)
92. IsOK = false;
93. else
94. IsOK = true;
95. }
96. // метод для расчёта градусной меры угла по теореме косинусов
97. public double CalculateAngle(double a, double b, double c) // метод расчёта одного угла
98. {
99. double angleInRadians = Math.Acos((b \* b + c \* c - a \* a) / (2 \* b \* c));
100. double angleInDegrees = RadiansToDegree(angleInRadians);
101. return angleInDegrees;
102. }
103. // метод преобразования радиан в градусы
104. private double RadiansToDegree(double rad) // метод преобразования радиан в градусы
105. {
106. return 180 \* rad / Math.PI;
107. }
108. }
109. private void CalculateWithoutAttributes\_Click(object sender, EventArgs e) // инициализация треугольника без параметров
110. {
111. Triangle triangle = new Triangle();
112. triangle.SideA = Convert.ToDouble(sideAInput);
113. triangle.SideB = Convert.ToDouble(sideBInput);
114. triangle.SideC = Convert.ToDouble(sideCInput);
115. triangle.Correct();
116. if (!triangle.IsOK)
117. {
118. MessageBox.Show(text: "Данные не корректны.\nРасчёт отменён", caption: "Ошибка 1", buttons: MessageBoxButtons.OK, MessageBoxIcon.Error);
119. return;
120. }
121. triangle.AngleA = triangle.CalculateAngle(triangle.SideA, triangle.SideB, triangle.SideC);
122. triangle.AngleB = triangle.CalculateAngle(triangle.SideB,triangle.SideA, triangle.SideC);
123. triangle.AngleC = triangle.CalculateAngle(triangle.SideC,triangle.SideA, triangle.SideB);
124. outputAngleA.Text = triangle.AngleA.ToString();
125. outputAngleB.Text = triangle.AngleB.ToString();
126. outputAngleC.Text = triangle.AngleC.ToString();
127. }
128. private void CalculateWIthAttributes\_Click(object sender, EventArgs e) // инициализация треугольника с параметрами
129. {
130. double A = Convert.ToDouble(sideAInput);
131. double B = Convert.ToDouble(sideBInput);
132. double C = Convert.ToDouble(sideCInput);
133. Triangle triangle = new Triangle(A, B, C);
134. if (!(triangle.IsOK))
135. {
136. MessageBox.Show(text: "Данные не корректны.\nРасчёт отменён", caption: "Ошибка 1", buttons: MessageBoxButtons.OK, MessageBoxIcon.Error);
137. return;
138. }
139. triangle.AngleA = triangle.CalculateAngle(triangle.SideA, triangle.SideB, triangle.SideC);
140. triangle.AngleB = triangle.CalculateAngle(triangle.SideB, triangle.SideA, triangle.SideC);
141. triangle.AngleC = triangle.CalculateAngle(triangle.SideC, triangle.SideA, triangle.SideB);
142. outputAngleA.Text = triangle.AngleA.ToString();
143. outputAngleB.Text = triangle.AngleB.ToString();
144. outputAngleC.Text = triangle.AngleC.ToString();
145. }
146. }
147. }

**1.2. Результаты работы программы**

Для проверки работоспособности программы возьмём треугольник со сторонами 9, 10, 11, заведомо зная, что такой треугольник существует, поэтому программа без нареканий выполняет расчёт углов с обоими интерфейсами (см. рисунок 1).

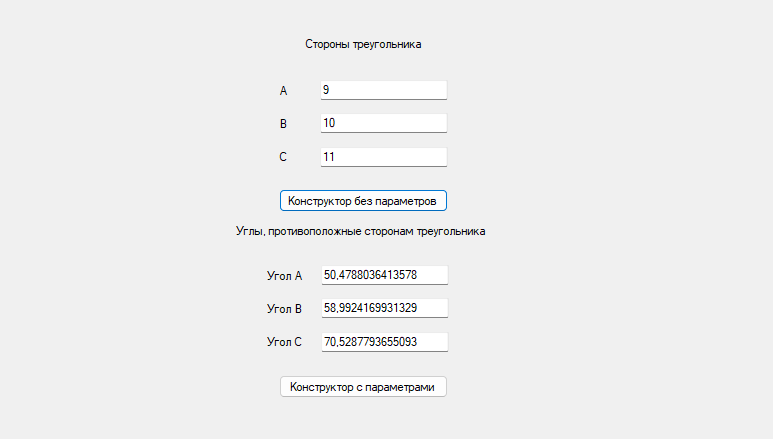


Рисунок 1 – Успешный расчёт углов

Теперь введём стороны треугольника, которого в реальности существовать не может. Пусть такими сторонами будут 5, 5, 15 (сумма первых двух меньше третей). В результате при попытке расчёта с каждым из интерфейсов программа выдаёт нам ошибку, связанную с некорректностью данных (см. рисунок 2).

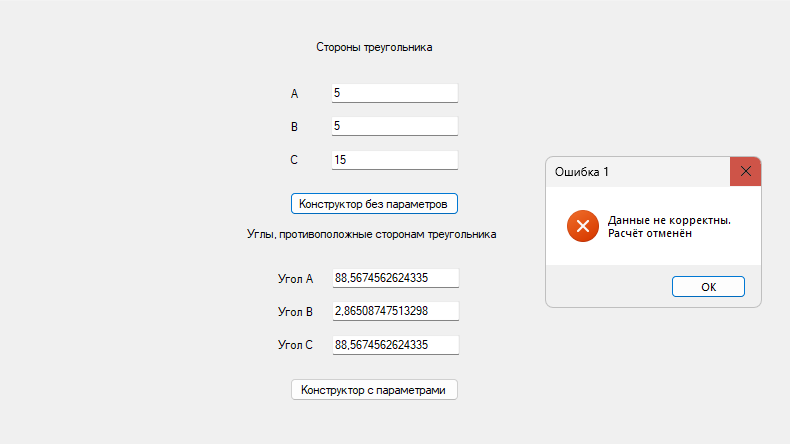


Рисунок 2 – Ошибка при расчёте

**1.3. Вывод**

Вариант реализации данной задачи на основе использования объектно-ориентированного программирования демонстрирует результаты, эквивалентные результатам алгоритмической реализации данной задачи при одинаковых входных данных (см. рисунок 3).

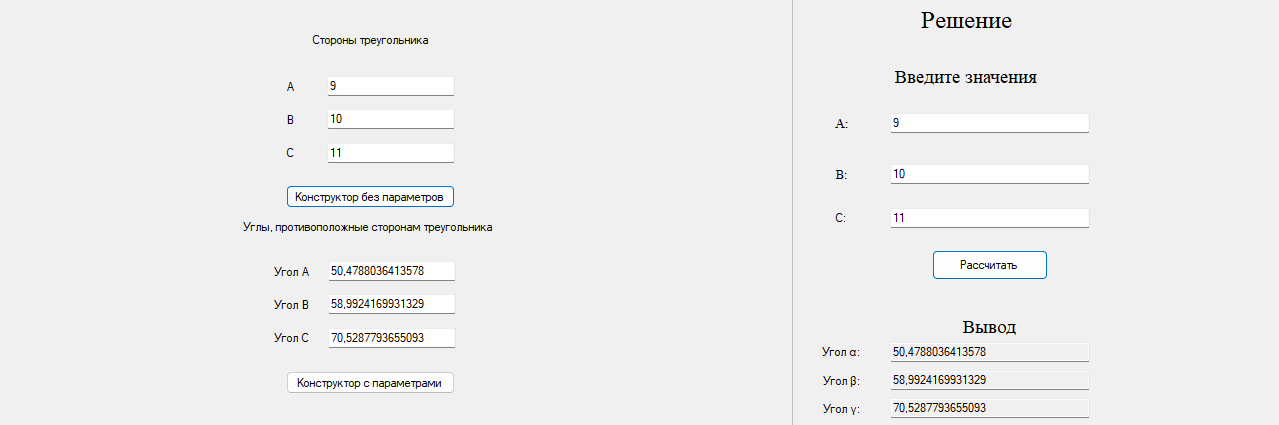


Рисунок 3 – Сравнение результатов

Также стоит отметить, что в предыдущем решении не было проверки корректности введённых данных, поэтому программа выводила “не число” в текстовые окна при попытке расчёта нереального треугольника (см. рисунок 4).

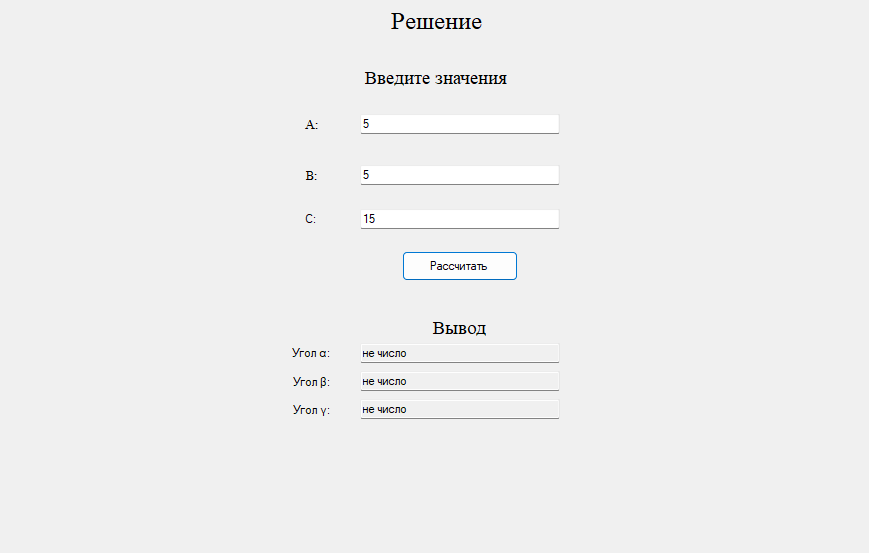


Рисунок 4 – Расчёт с неверными данными

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 7.32-2017. Межгосударственный стандарт. Система стандартов по информации, библиотечному и издательскому делу. Отчет о научно-исследовательской работе. Структура и правила оформления" (введен в действие Приказом Росстандарта от 24.10.2017 N 1494-ст) (дата обращений 01.10.2023).