**Лабораторная работа №2**

**«Программная реализация метода пошаговой детализации»**

**1.Задание на разработку проекта**

Создать приложение для вычисления длин медиан, проведенных ко всем сторонам треугольника, зная длины сторон треугольника.

Задано: a, b, c – длины стороны

Требуется определить:

med\_a – медиану к стороне а;

med\_b – медиана к стороне b;

med\_c – медиана к стороне c;

Ограничение на значения исходных данных и их соотношение:

a > 0; b > 0; c > 0;

a + b > c;

a + c > b;

b + c > a;

**2.Формализация и уточнение задания**

Для решения задачи существуют известные формулы:

(1)

Где a, b, c – стороны треугольника.

В соответствии с требованиями задания разобьем решение задачи на несколько отдельных задач и создадим подпрограммы пользователя, реализующие эти задачи:

* Метод double Med( ) вычисляет медианы, проведенные к сторонам;
* Метод Out( ), созданный для вписывания данных в double Med();
* Метод double Vvod( ), созданный для ввода значений с ***TextBox***;
* Метод void Vivod(), созданный для вывода значений в ***TextBox***.

**3.Разработка приложения**

3.1.Разработка графического интерфейса пользователя

После тщательного анализа выяснилось, что для разработки проекта нам понадобится 3 элемента для ввода значений и 3 элемента для вывода значений, а именно ***TextBox*** и 2 кнопки ***button***, одна для выхода, другая для вычисления. Также, следует разместить 6 элементов ***label*** для удобства пользователя, чтобы он понимал куда записывать входные значения и где можно увидеть выходные значения. Таким образом форма может выглядеть как на рисунке 1.

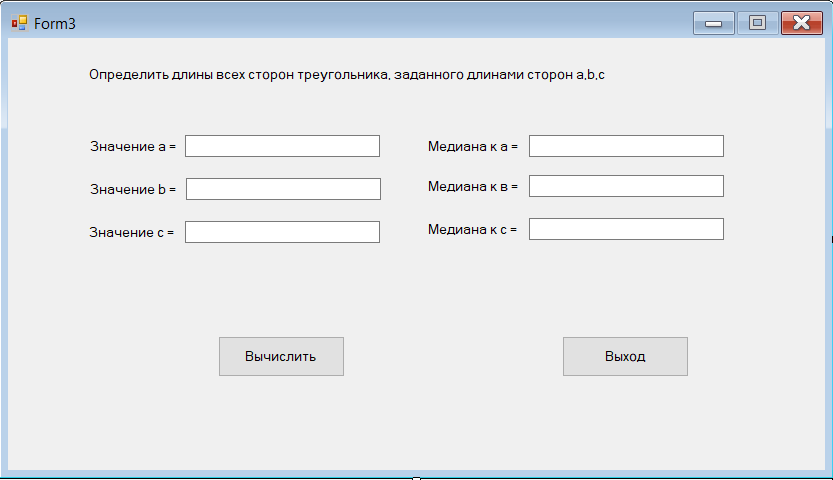


Рисунок 1 – Графический интерфейс пользователя

3.2.Разработка схем алгоритмов

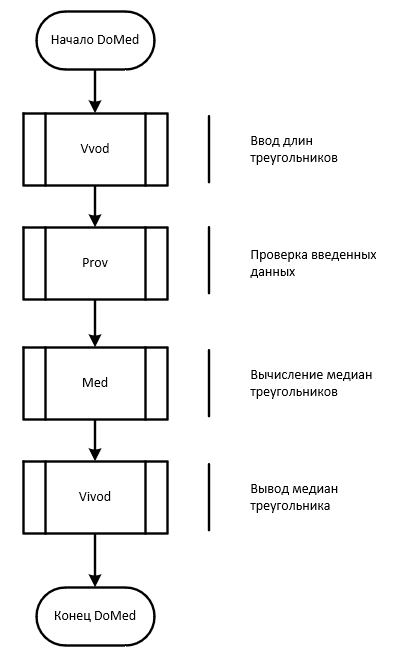


Рисунок 2 – Схема алгоритмов процедур

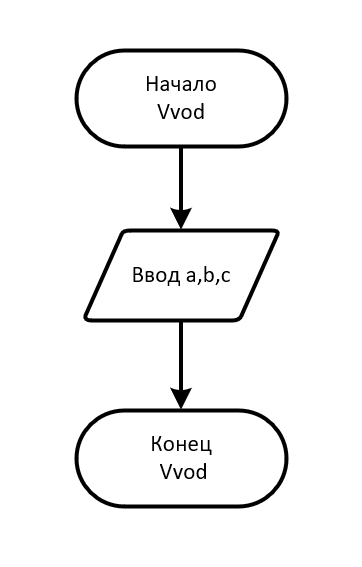


Рисунок 3 – Схема алгоритмов процедур Vvod

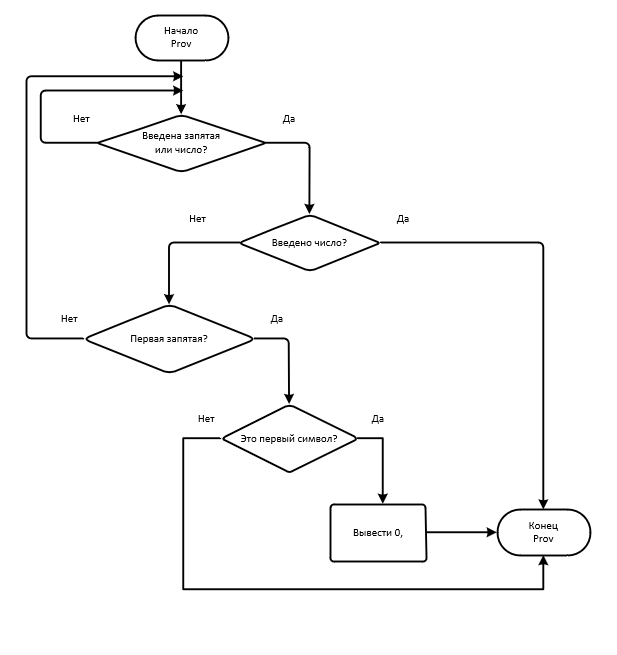


Рисунок 4 –Схема проверки введенных данных

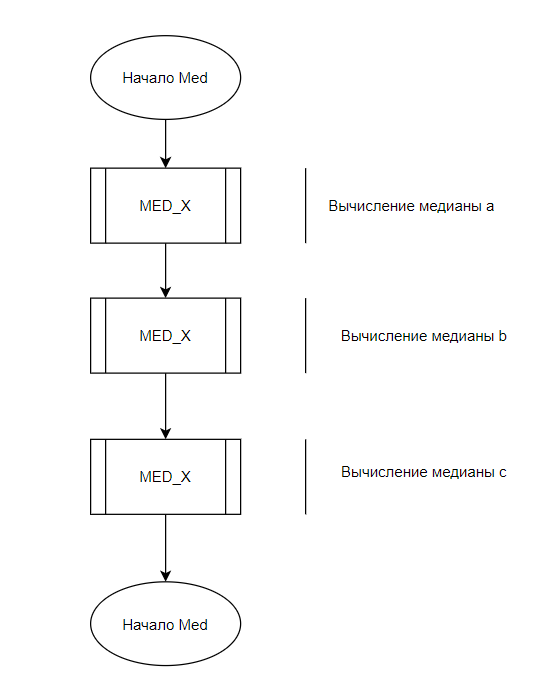


Рисунок 5 – Схема алгоритмов процедур Med

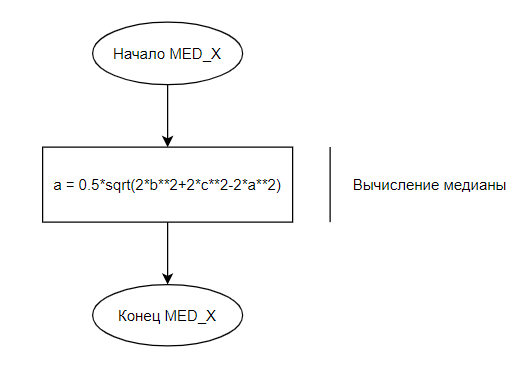


Рисунок 6 – Схема алгоритмов процедур MED\_X

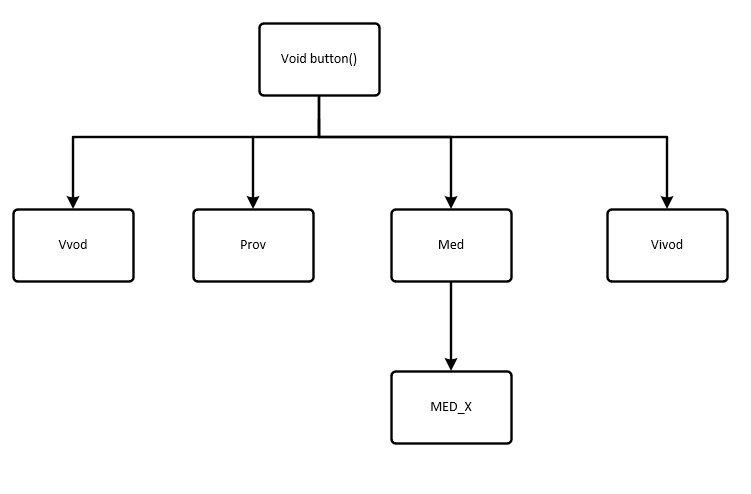


Рисунок 7 – Схема иерархии процедур для решения задачи

**4.Разработка программного кода**

Все подпрограммы-методы размещаются в классе (по умолчанию Class1).

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Globalization;

namespace AnP\_proj

{

internal class Class1

{

// метод вводa в TextBox

public static double Vvod(TextBox textbox)

{

return Convert.ToDouble(textbox.Text);

}

// Вывод в TextBox

public static void Vivod(TextBox textBox, double c)

{

textBox.Text = Convert.ToString(c);

}

//метод вывода

public static void Med(double a, double b, double c, ref double med\_a, ref double med\_b, ref double med\_c)

{

MED\_X(a, b, c, out med\_a);

MED\_X(b,c,a , out med\_b);

MED\_X(c,a, b, out med\_c);

}

// метод вычисления медианы

public static double MED\_X(double a, double b, double c, out double med\_a)

{

//double med\_a, med\_b, med\_c;

med\_a = 0.5 \* Math.Sqrt(2 \* Math.Pow(b, 2) + 2 \* Math.Pow(c, 2) - Math.Pow(a, 2));

return 0;

}

}

}

В основной части проекта подпрограммы-методы вызываются из class 1

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Globalization;

namespace AnP\_proj

{

public partial class Form3 : Form

{

int x1;

char ds = (char)NumberFormatInfo.CurrentInfo.NumberDecimalSeparator[0];

public Form3()

{

InitializeComponent();

}

// реализация вызова метода с модификатором ref

private void button1\_Click(object sender, EventArgs e)

{

//double P = 0;

double med\_a, med\_b, med\_c;

med\_a = med\_b = med\_c = 0;

double a = Class1.Vvod(textBox1);

double b = Class1.Vvod(textBox2);

double c = Class1.Vvod(textBox3);

if (a <= 0)

{

MessageBox.Show("Указанное значение переменной 'a' выходит за пределы области допустимых значений!", "Ошибка!");

return;

}

if (b <= 0)

{

MessageBox.Show("Указанное значение переменной 'b' выходит за пределы области допустимых значений!", "Ошибка!");

return;

}

if (c <= 0)

{

MessageBox.Show("Указанное значение переменной 'c' выходит за пределы области допустимых значений!", "Ошибка!");

return;

}

if ((a + b <= c) || (a + c <= b) || (c + b <= a))

{

MessageBox.Show("Указанное значение переменных выходит за пределы области допустимых значений!", "Ошибка!");

}

//Class1.P1(n, ref P);

//Class1.Vivod(textBox2, P);

Class1.Med(a, b, c, ref med\_a, ref med\_b,ref med\_c);

Class1.Vivod(textBox4, med\_a);

Class1.Vivod(textBox5, med\_b);

Class1.Vivod(textBox6, med\_c);

}

private void button2\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void textBox1\_KeyPress(object sender, KeyPressEventArgs e)

{

// Если нажатая клавиша не цифровая.

if (!Char.IsDigit(e.KeyChar))

// Запрет на ввод более одного десятичного разделителя.

if (e.KeyChar != ds || textBox1.Text.IndexOf(ds) != -1)

// Если нажатая клавиша не является клавишей BackSpace.

if (e.KeyChar != (char)Keys.Back)

e.Handled = true; // Запрет ввода

// Запрет ввода повторных нулей в начале числа.

if (e.KeyChar == '-')

e.Handled = true;

if (e.KeyChar == (char)Keys.D0 || e.KeyChar == (char)Keys.NumPad0)

if (textBox1.Text.Length >= 1) // Если строка не пустая

if (textBox1.Text[0] == '0' && textBox1.SelectionStart < 2)

e.Handled = true; // Запрет ввода

//-незначащий ноль

if (Char.IsDigit(e.KeyChar))

if (textBox1.Text.Length == 1 && textBox1.Text[0] == '0')

textBox1.Paste(",");

// Замена десятичного разделителя в начале числа на "0,".

if (e.KeyChar == ds)

{ // Если введен десятичный разделитель

x1 = 0;

if (textBox1.Text != "" && textBox1.Text[0] == '-') { x1 = 1; }

if (textBox1.Text.IndexOf(ds) == -1 && textBox1.SelectionStart == x1)

{

// Если десятичный разделитель уже есть, то 0 не вставлять.

textBox1.Text = textBox1.Text.Insert(x1, "0"); // Вставка 0

textBox1.SelectionStart = x1 + 1; // Текстовый курсор в конец

}

// Перенос точки ввода минуса в начало (ввод минуса только спереди)

}

if (Char.IsDigit(e.KeyChar))

if (textBox1.Text.Length == 1 && textBox1.Text[0] == '0')

textBox1.Paste(",");

}

private void label5\_Click(object sender, EventArgs e)

{

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

}

private void textBox5\_TextChanged(object sender, EventArgs e)

{

}

private void textBox2\_KeyPress(object sender, KeyPressEventArgs e)

{

// Если нажатая клавиша не цифровая.

if (!Char.IsDigit(e.KeyChar))

// Запрет на ввод более одного десятичного разделителя.

if (e.KeyChar != ds || textBox2.Text.IndexOf(ds) != -1)

// Если нажатая клавиша не является клавишей BackSpace.

if (e.KeyChar != (char)Keys.Back)

e.Handled = true;

if (e.KeyChar == '-')

e.Handled = true;

if (e.KeyChar == (char)Keys.D0 || e.KeyChar == (char)Keys.NumPad0)

if (textBox2.Text.Length >= 1) // Если строка не пустая

if (textBox2.Text[0] == '0' && textBox2.SelectionStart < 2)

e.Handled = true; // Запрет ввода

//-незначащий ноль

if (Char.IsDigit(e.KeyChar))

if (textBox2.Text.Length == 1 && textBox2.Text[0] == '0')

textBox2.Paste(",");

// Замена десятичного разделителя в начале числа на "0,".

if (e.KeyChar == ds)

{ // Если введен десятичный разделитель

x1 = 0;

if (textBox2.Text != "" && textBox2.Text[0] == '-') { x1 = 1; }

if (textBox2.Text.IndexOf(ds) == -1 && textBox2.SelectionStart == x1)

{

// Если десятичный разделитель уже есть, то 0 не вставлять.

textBox2.Text = textBox2.Text.Insert(x1, "0"); // Вставка 0

textBox2.SelectionStart = x1 + 1; // Текстовый курсор в конец

}

// Перенос точки ввода минуса в начало (ввод минуса только спереди)

}

if (Char.IsDigit(e.KeyChar))

if (textBox2.Text.Length == 1 && textBox2.Text[0] == '0')

textBox2.Paste(",");

}

private void textBox3\_KeyPress(object sender, KeyPressEventArgs e)

{

// Если нажатая клавиша не цифровая.

if (!Char.IsDigit(e.KeyChar))

// Запрет на ввод более одного десятичного разделителя.

if (e.KeyChar != ds || textBox3.Text.IndexOf(ds) != -1)

// Если нажатая клавиша не является клавишей BackSpace.

if (e.KeyChar != (char)Keys.Back)

e.Handled = true; // Запрет ввода

// Запрет ввода повторных нулей в начале числа.

if (e.KeyChar == '-')

e.Handled = true;

if (e.KeyChar == (char)Keys.D0 || e.KeyChar == (char)Keys.NumPad0)

if (textBox3.Text.Length >= 1) // Если строка не пустая

if (textBox3.Text[0] == '0' && textBox3.SelectionStart < 2)

e.Handled = true; // Запрет ввода

//-незначащий ноль

if (Char.IsDigit(e.KeyChar))

if (textBox3.Text.Length == 1 && textBox3.Text[0] == '0')

textBox3.Paste(",");

// Замена десятичного разделителя в начале числа на "0,".

if (e.KeyChar == ds)

{ // Если введен десятичный разделитель

x1 = 0;

if (textBox3.Text != "" && textBox3.Text[0] == '-') { x1 = 1; }

if (textBox3.Text.IndexOf(ds) == -1 && textBox3.SelectionStart == x1)

{

// Если десятичный разделитель уже есть, то 0 не вставлять.

textBox3.Text = textBox3.Text.Insert(x1, "0"); // Вставка 0

textBox3.SelectionStart = x1 + 1; // Текстовый курсор в конец

}

// Перенос точки ввода минуса в начало (ввод минуса только спереди)

}

if (Char.IsDigit(e.KeyChar))

if (textBox3.Text.Length == 1 && textBox3.Text[0] == '0')

textBox3.Paste(",");

}

}

}

**5.Выполнение приложения и получение результатов**

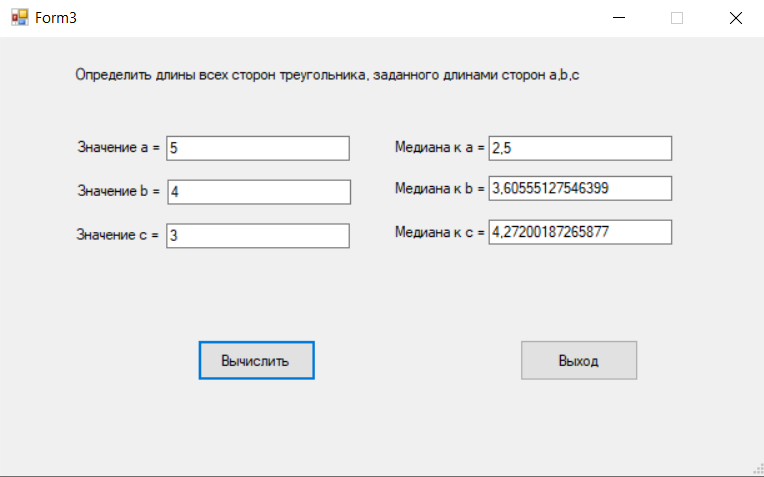
****

Рисунок 8 – Расчет Медиан №1

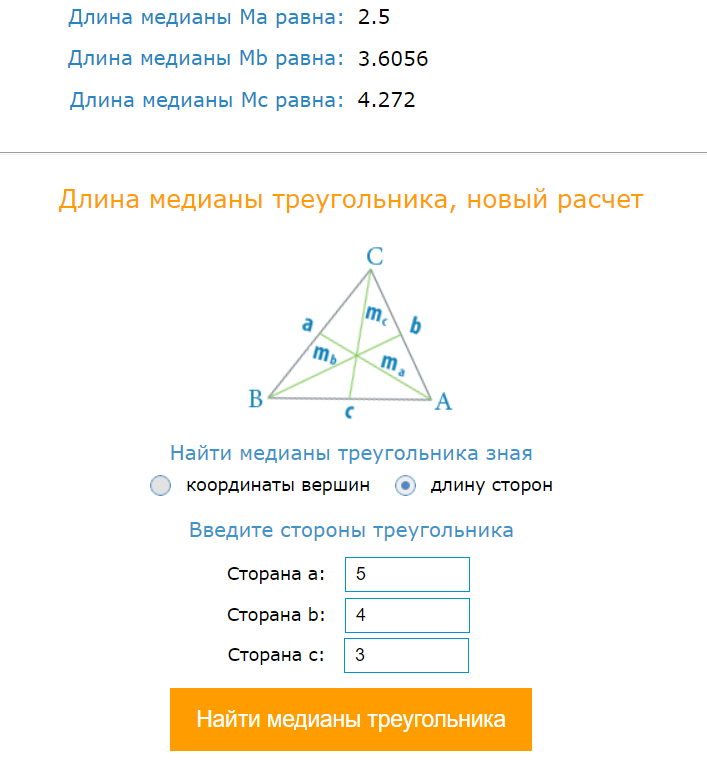


Рисунок 9 - Проверка результата №1

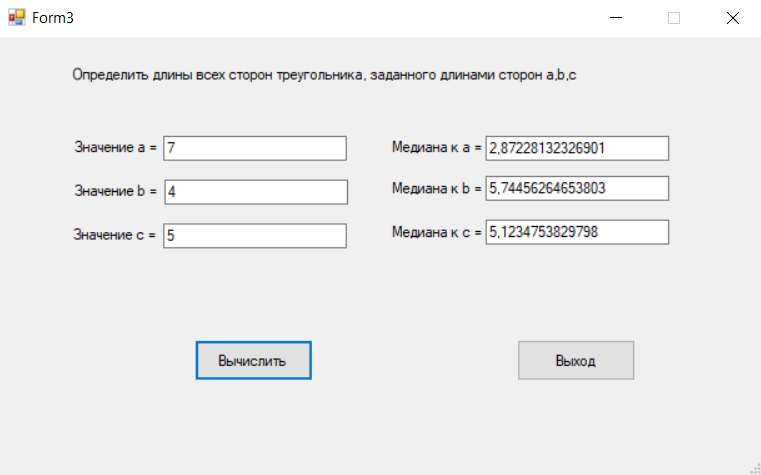


Рисунок 10 – Расчёт медиан №2

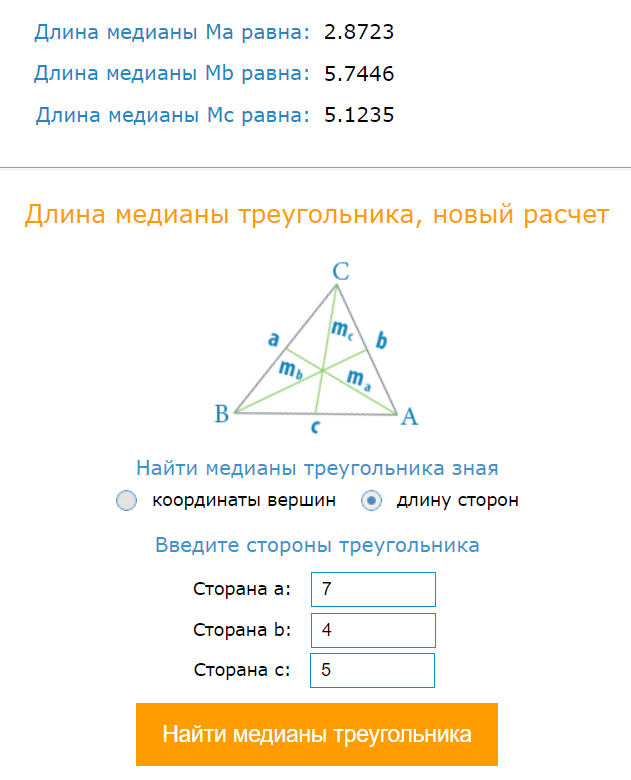


Рисунок 11 – Проверка Результата № 2