**Лабораторная работа №4**

**«Циклический алгоритм»**

**1. Постановка задачи**

Решить задачу формирования таблицы значений функции. При изменении x на отрезке [a; b] с шагом h.

(1)

**2. Формализация и уточнение задачи**

Будем считать, что исходные данные и результаты вычислений будут иметь тип ***double***. Так как заданы границы цикла и шаг изменения аргумента, то мы можем определить количество точек табуляции по формуле:

(2)

Где a и b – границы отрезка табуляции, h – шаг табуляции, а квадратные скобки означают взятие целой части.

Алгоритм решения задачи относится к регулярным циклам с параметром, так как количество повторений известно и программа повторяет одни и те же действия.

Для реализации задания разработаем следующие функциональные алгоритмы:

* ***Vvod()***,выполняет вход исходных данных;
* ***Vivod()***, выполняет вывод конечной функции*;*
* ***VivodDGV()****,* выполняется вывод в таблицу каждого значения;
* ***Func()****,* выполняет вычисление функции;
* ***Tab()****,* просчитывает каждый шаг;
* ***Prof()****,* проверяет введенные данные;

**3. Разработка проекта**

3.1. Разработка графического интерфейса пользователя

Интерфейс пользователя предоставлен на рисунке 1.

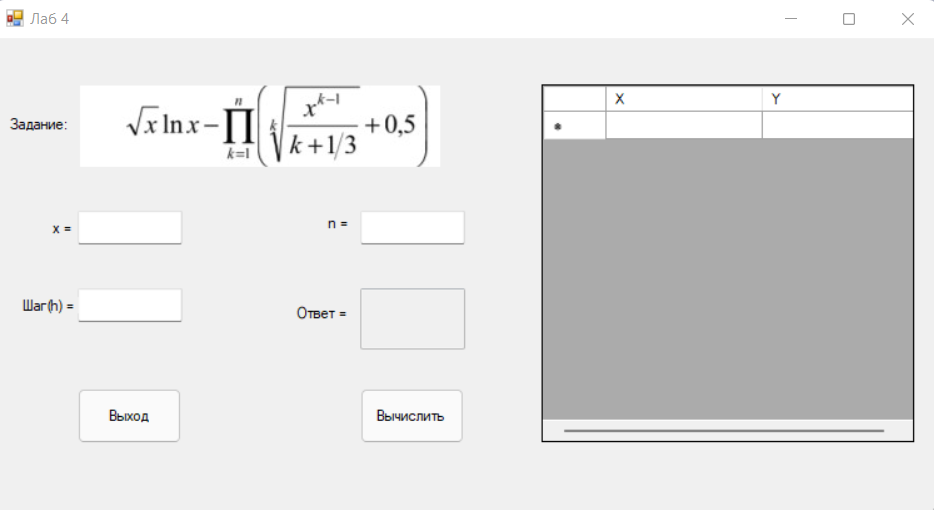


Рисунок 1 – Графический интерфейс пользователя

3.2. Разработка схемы алгоритмов

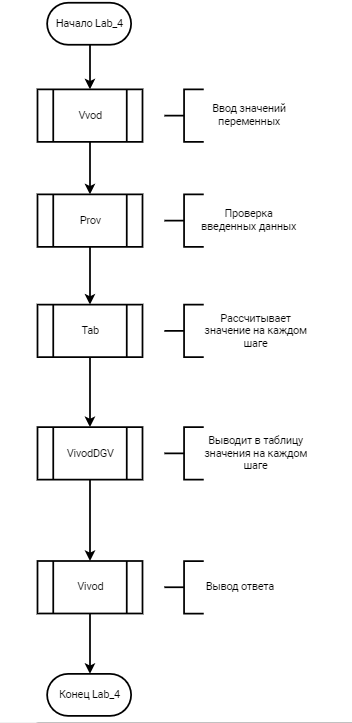
**

Рисунок 2 – схема алгоритмов процедур

Проверка вводимых элементов будет проводиться с помощью данной функции

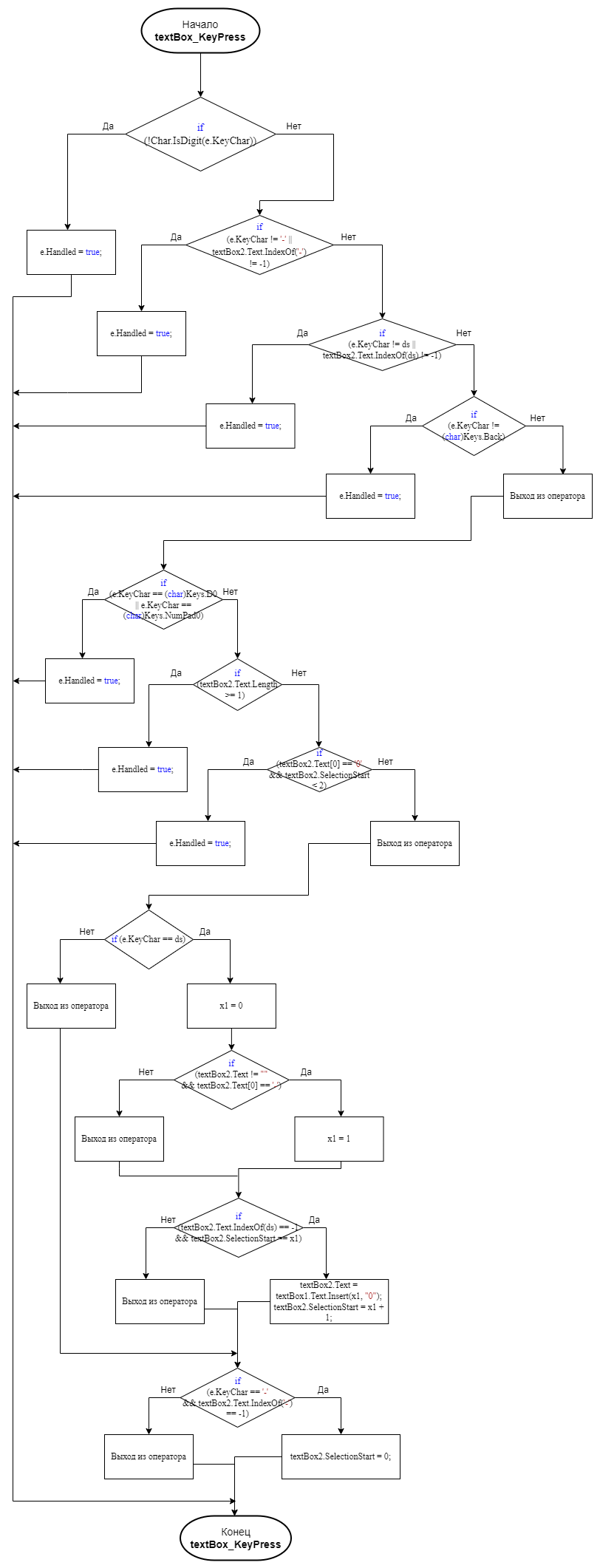


Рисунок 3 – проверка вводимых данных

Блок схемы функций из библиотеки 4Laba предоставлена на рисунках 4 – 8.

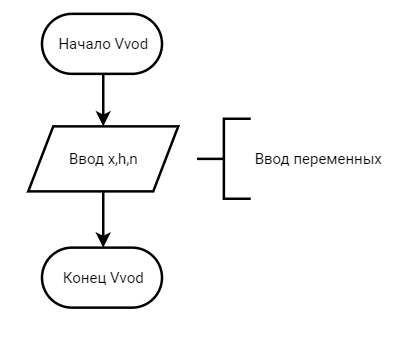
**

Рисунок 4 – Схема алгоритмов процедур **Vvod()**

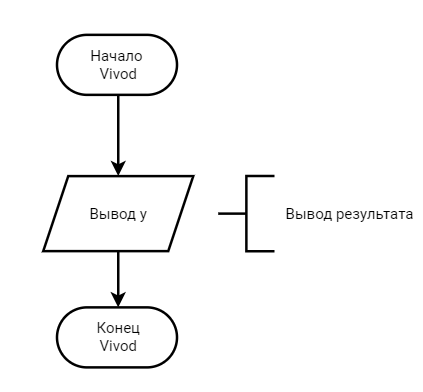
**

Рисунок 5 - Схема алгоритмов процедур **Vivod()**

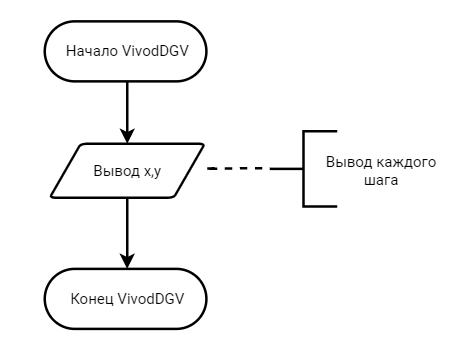
****

Рисунок 6 - Схема алгоритмов процедур **VivodDGV()**

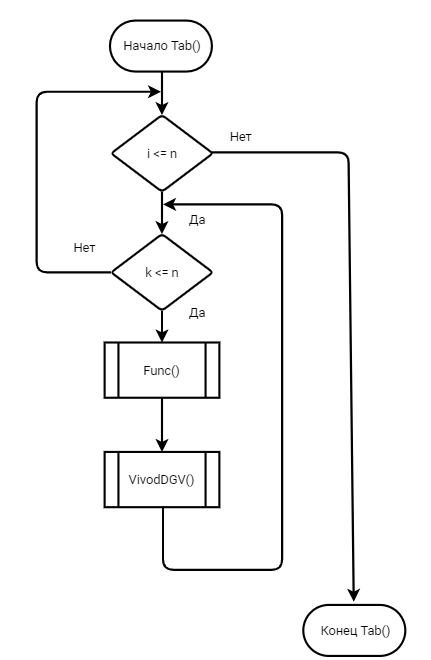


Рисунок 7 - Схема алгоритмов процедур **Tab()**

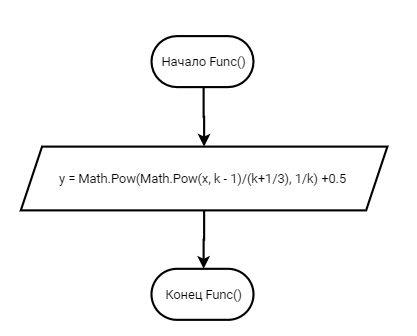


Рисунок 8 - Схема алгоритмов процедур **Func()**

3.2. Написание программного кода

Основной код:

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 Form5 : Form

{

public Form5()

{

InitializeComponent();

}

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

{

Application.Exit();

}

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

{

if (textBox1.Text == "" || textBox2.Text == "" || textBox3.Text == "")

{

MessageBox.Show(this,

"Введены пустые значения", "Лабораторная работа 4",

MessageBoxButtons.OKCancel,

MessageBoxIcon.Exclamation,

MessageBoxDefaultButton.Button1);

return;

}

if (Convert.ToDouble(textBox1.Text) < 0 || Convert.ToDouble(textBox2.Text) < 0 || Convert.ToDouble(textBox3.Text) < 0)

{

MessageBox.Show(this,

"Шаг не может быть отрицательным",

"Лабораторная работа 4",

MessageBoxButtons.OKCancel,

MessageBoxIcon.Exclamation,

MessageBoxDefaultButton.Button1);

return;

}

dataGridView1.Rows.Clear();

double x = Dll.Class4.Vvod(textBox1);

double n = Dll.Class4.Vvod(textBox2);

double h = Dll.Class4.Vvod(textBox3);

Dll.Class4.Tab(x,n,h, dataGridView1, out double sum, out double y);

Dll.Class4.Vivod(textBox4, y);

}

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

{

Dll.Class4.Prof(textBox1, e);

}

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

{

Dll.Class4.Prof(textBox2, e);

}

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

{

Dll.Class4.Prof(textBox3, e);

}

}

}

Все подпрограммы-методы размещаются в классе (4Laba).

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Globalization;

//using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Dll

{

public class Class4

{

public static void Vivod(TextBox t, double c)

{

t.Text = c.ToString("F");

}

public static double Vvod(TextBox textBox)

{

if (textBox.Text == "")

{

MessageBox.Show("Некоторые знгачения пустые", "Ошибка", MessageBoxButtons.OKCancel);

return 0;

}

return Convert.ToDouble(textBox.Text);

}

public static void VivodDGV(DataGridView DGV, double x, double y)

{

DGV.Rows.Add(x.ToString("F1"), y.ToString("F5"));

}

public static double Func(double x, double k)

{

double y;

y = Math.Pow(Math.Pow(x, k - 1)/(k + 1/3), 1/k) + 0.5;

return y;

}

/// <summary>

/// Метод для вычиления значения функции и записи в таблицу

/// </summary>

/// <param name="a">Начало</param>

/// <param name="b">Конец</param>

/// <param name="h>">Шаг</param>

/// <param name="DGV">Таблица DVG</param>

/// <param name="sum">Вывод числа y</param>

public static void Tab(double a, double b, double h, DataGridView DGV, out double sum, out double y)

{

double x;

x = a;

int n = Convert.ToInt32(Math.Round((b – a) /h + 1));

sum = 1;

for (int I = 1; I <= n; i++)

{

for (int k = 1; k <= n; k++)

{

sum += Func(x, k);

}

VivodDGV(DGV, x, sum);

x += h;

}

y = Math.Sqrt(x) \* Math.Log(x) – sum;

}

public static void Prof(TextBox textBox1, KeyPressEventArgs e)

{

int x1;

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

// Если нажатая клавиша не цифровая.

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

// Запрет на ввод более одного знака минуса.

If (e.KeyChar != ‘-‘ || textBox1.Text.IndexOf(‘-‘) != -1)

// 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 == (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 (e.KeyChar == ‘-‘ && textBox1.Text.IndexOf(‘-‘) == -1)

textBox1.SelectionStart = 0; // Текстовый курсор в начало

//-незначащий ноль

}

}

}

**4. Создание XML-документации**

Ниже предоставлен код XML-документации.

<?xml version=”1.0”?>

<doc>

<assembly>

<name>Dll</name>

</assembly>

<members>

<member name=”M:Dll.Class4.Tab(System.Double,System.Double,System.Double,System.Windo ws.Forms.DataGridView,System.Double@,System.Double@)”>

<summary>

Метод для вычиления значения функции и записи в таблицу

</summary>

<param name=”a”>Начало</param>

<param name=”b”>Конец</param>

<param name=”h>>”>Шаг</param>

<param name=”DGV”>Таблица DVG</param>

<param name=”sum”>Вывод числа y</param>

</member>

<member name=”F:Dll.Form1.components”>

<summary>

Required designer variable.

</summary>

</member>

<member name=”M:Dll.Form1.Dispose(System.Boolean)”>

<summary>

Clean up any resources being used.

</summary>

<param name=”disposing”>true if managed resources should be disposed; otherwise, false.</param>

</member>

<member name=”M:Dll.Form1.InitializeComponent”>

<summary>

Required method for Designer support – do not modify

the contents of this method with the code editor.

</summary>

</member>

</members>

</doc>

**5. Результаты выполнения проекта**

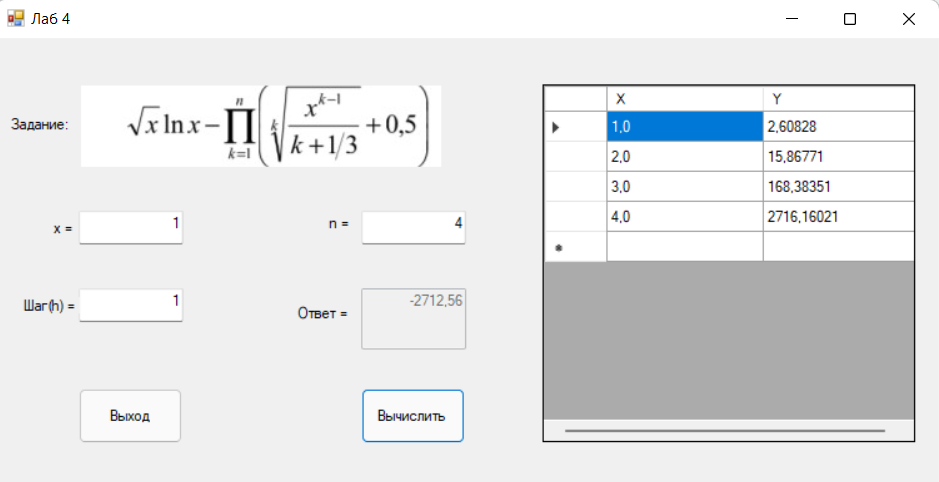
****

Рисунок 9 – Результат выполнения программы