**МИНИСТЕРСТВО ТРАНСПОРТА**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ**

**АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО**

**ОБРАЗОВАНИЯ**

**“РОССИЙСКИЙ УНИВЕРСИТЕТ**

**ТРАНСПОРТА” (МИИТ)**

**Институт транспортной техники и систем**

**управления**

**(ИТТСУ)**

**Кафедра “Управление и защита информации”**

**Тема: “Лабораторная работа №3”**

**Выполнили студенты группы ТКИ-342**

**Павлючков Даниил Дмитриевич**

**Рудов Лев Андреевич**

**Специальность 10.05.01 “Компьютерная безопасность”**

**Преподаватель: к.т.н. Сафронов А.И.**

**Москва**

**2024**

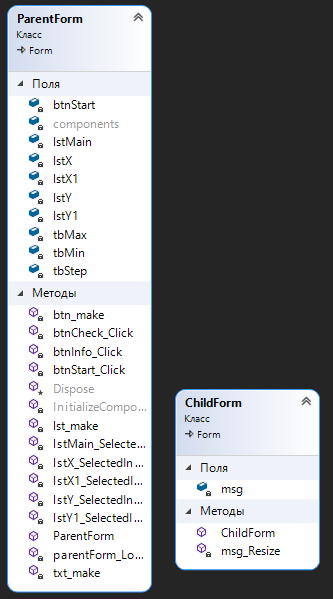
1. Цель работы:

Закрепить навыки разработки визуального пользовательского интерфейса, освоить работу с текстовыми файлами в среде *Microsoft Visual Studio*, научиться взаимно увязывать одни элементы управления с другими, получить представление о перерисовке и перемасштабировании

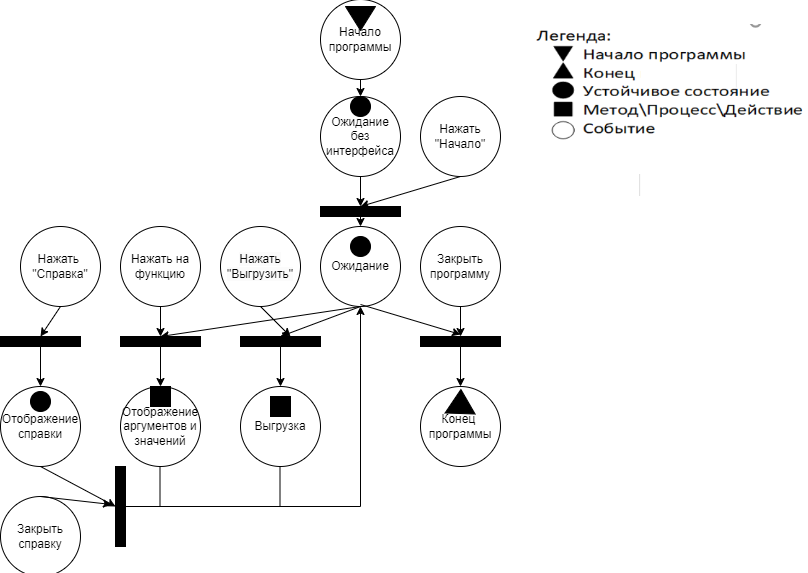
2. Задача:

В интегрированной среде разработки *Microsoft Visual Studio* на языке *Visual C#* разработать графический пользовательский интерфейс, состоящий из двух экранных форм (основной и дочерней), все элементы управления которого полностью настраиваются кодом без использования графического конструктора».

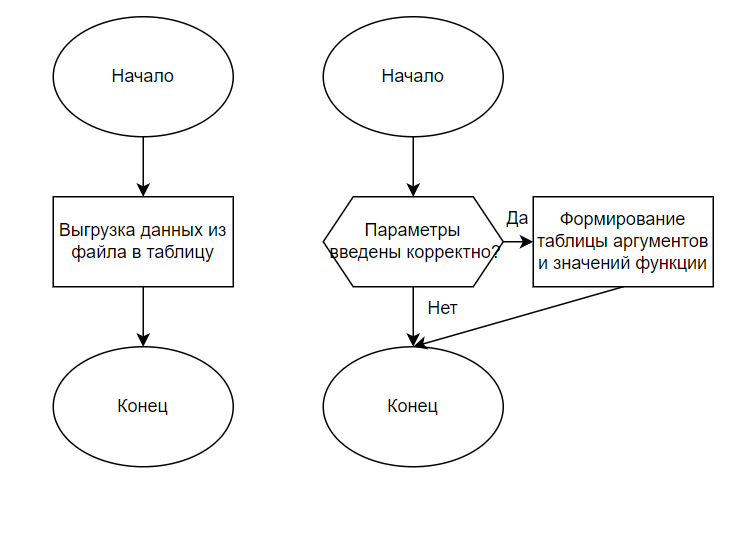
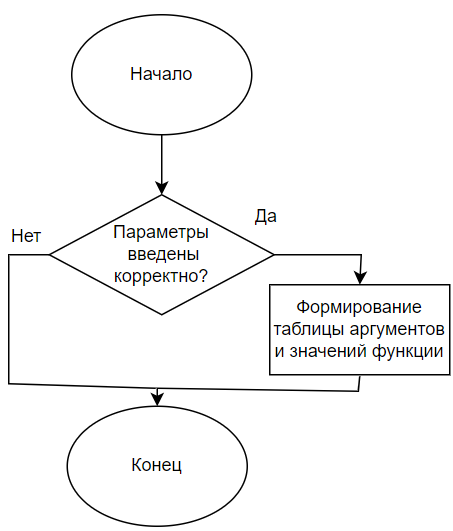
3. Диаграмма классов:



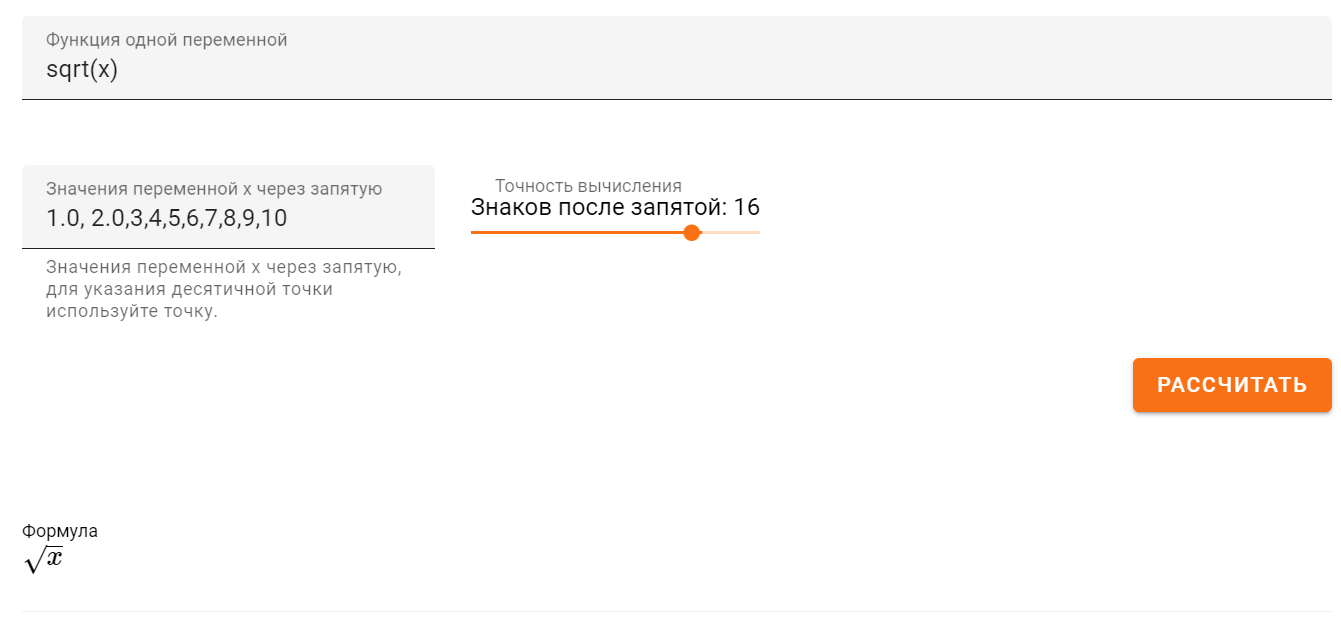
4. Сеть Петри

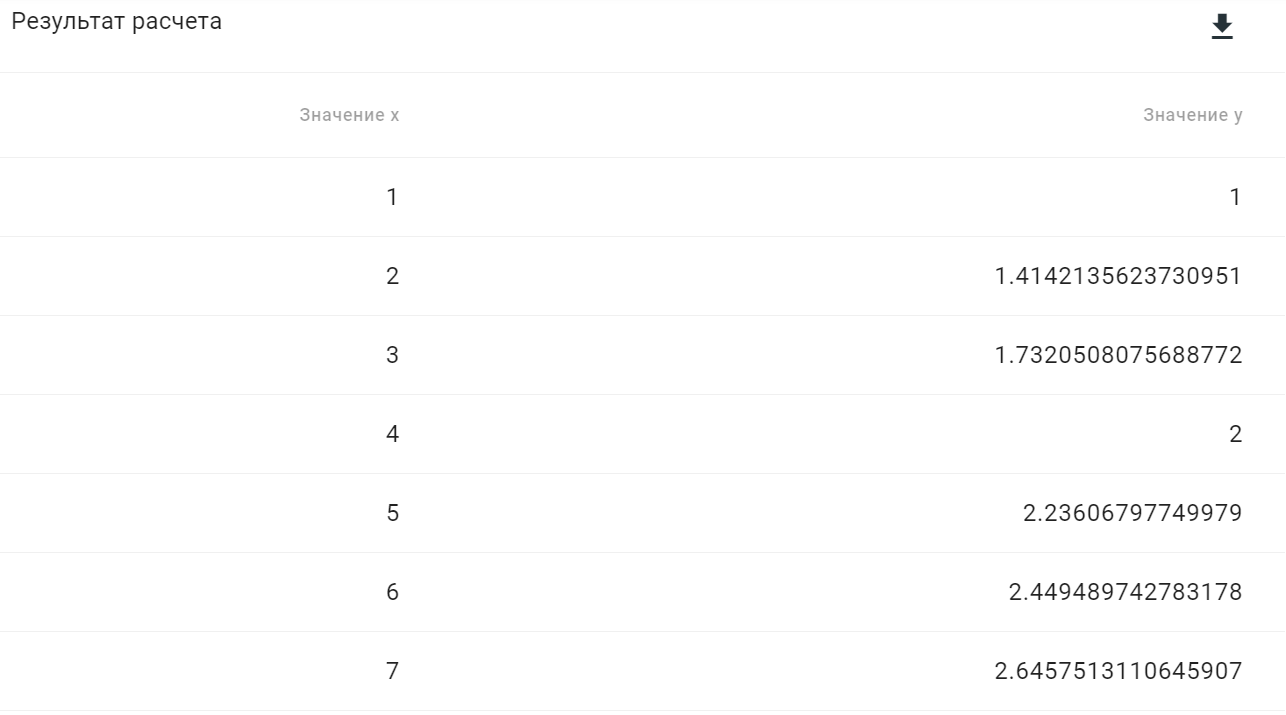


5. Схемы алгоритмов методов:



6. Подбор тестовых примеров





7. Листинг кода:

using System;

using System.Drawing;

using System.IO;

using System.Windows.Forms;

namespace Lab3

{

public partial class ParentForm : Form

{

public ParentForm()

{

InitializeComponent();

}

private Button btnStart;

private TextBox tbMin;

private TextBox tbMax;

private TextBox tbStep;

private ListBox lstY;

private ListBox lstX;

private ListBox lstX1;

private ListBox lstY1;

private ListBox lstMain;

private void parentForm\_Load(object sender, EventArgs e)

{

btn\_make();

txt\_make();

lst\_make();

btnStart.Visible = true;

}

private void btn\_make()

{

BackColor = SystemColors.Info;

Cursor = Cursors.Hand;

Button buttonGenerator(int x, int y, string label, bool anchor)

{

Button example = new Button();

example.Location = new Point(this.Width - x, this.Height - y);

example.Text = label;

example.AutoSize = true;

example.BackColor = Color.Black;

example.Padding = new Padding(6);

example.Font = new Font("Times New Roman", 18);

example.Visible = false;

example.ForeColor = Color.White;

if (anchor) example.Anchor = (AnchorStyles.Top | AnchorStyles.Right);

return example;

}

Button btnInfo = buttonGenerator(100, 485, "Справка", true);

btnInfo.Click += btnInfo\_Click;

Controls.Add(btnInfo);

btnStart = buttonGenerator(800, 100, "Нажмите, чтобы начать работу", false);

btnStart.Click += btnStart\_Click;

Controls.Add(btnStart);

Button btnCheck = buttonGenerator(100, 100, "Выгрузить", true);

btnCheck.Click += btnCheck\_Click;

Controls.Add(btnCheck);

}

private void txt\_make()

{

TextBox textBoxGenerator(int x, int y)

{

TextBox example = new TextBox();

example.Location = new Point(x, y);

example.AutoSize = true;

example.Visible = false;

return example;

}

tbMin = textBoxGenerator(btnStart.Left, this.Height - 470);

tbMin.Text = "Нижняя граница";

Controls.Add(tbMin);

tbStep = textBoxGenerator(btnStart.Left + btnStart.Width / 3, this.Height - 470);

tbStep.Text = "Шаг";

Controls.Add(tbStep);

tbMax = textBoxGenerator(btnStart.Left + 2 \* btnStart.Width / 3, this.Height - 470);

tbMax.Text = "Вехняя граница";

Controls.Add(tbMax);

}

private void lst\_make()

{

ListBox listBoxGenerator(int x, int y)

{

ListBox example = new ListBox();

example.Location = new Point(x, y);

example.Size = new Size(btnStart.Width / 3, 200);

example.Visible = false;

return example;

}

lstX = listBoxGenerator(btnStart.Left + btnStart.Width / 3, 65);

lstX.SelectedIndexChanged += lstX\_SelectedIndexChanged;

Controls.Add(lstX);

lstY = listBoxGenerator(lstX.Right, 65);

lstY.SelectedIndexChanged += lstY\_SelectedIndexChanged;

Controls.Add(lstY);

lstX1 = listBoxGenerator(lstY.Right, 65);

lstX1.Text = "Аргументы";

lstX1.SelectedIndexChanged += lstX1\_SelectedIndexChanged;

Controls.Add(lstX1);

lstY1 = listBoxGenerator(lstX1.Right, 65);

lstY1.Text = "Функции";

lstY1.SelectedIndexChanged += lstY1\_SelectedIndexChanged;

Controls.Add(lstY1);

lstMain = listBoxGenerator(btnStart.Left, 65);

lstMain.Items.Add("Sin (x)");

lstMain.Items.Add("Log\_2 (x)");

lstMain.Items.Add("Cbrt(x^2) / (x^2 - 1)");

lstMain.Items.Add("ArcCos (x)");

lstMain.Items.Add("Sqrt (x)");

lstMain.Items.Add("Cos (x)");

lstMain.Items.Add("ArcSin (x)");

lstMain.Items.Add("Log\_e (x)");

lstMain.Items.Add("Log\_10 (x)");

lstMain.Items.Add("ArcTg (x)");

lstMain.Items.Add("x^2");

lstMain.Items.Add("Tan (x)");

lstMain.SelectedIndexChanged += lstMain\_SelectedIndexChanged;

Controls.Add(lstMain);

}

private void lstMain\_SelectedIndexChanged(object sender, EventArgs e)

{

if (lstMain.SelectedIndex > -1)

{

lstX.Items.Clear();

lstY.Items.Clear();

if (string.IsNullOrEmpty(tbMin.Text) || string.IsNullOrEmpty(tbMax.Text))

{

return;

}

bool successMin = double.TryParse(tbMin.Text, out double resultMin);

bool successStep = double.TryParse(tbStep.Text, out double resultStep);

bool successMax = double.TryParse(tbMax.Text, out double resultMax);

if (!successMin || !successStep || !successMax )

{

return;

}

int counter = 0;

using (StreamWriter sw = new StreamWriter("Data.txt"))

{

double arg = Double.Parse(tbMin.Text);

double setstep = Double.Parse(tbStep.Text);

int top = Convert.ToInt32(Math.Ceiling((Double.Parse(tbMax.Text) - Double.Parse(tbMin.Text))));

int convstep = Convert.ToInt32(Math.Ceiling(Double.Parse(tbStep.Text)));

for (double i = arg; i <= top + 1; i = i+convstep)

{

lstX.Items.Add(arg);

switch (lstMain.SelectedIndex)

{

case 0:

lstY.Items.Add(Math.Sin(i));

break;

case 1:

lstY.Items.Add(Math.Log(i, 2));

break;

case 2:

if (i > 0.9999999 && i < 1.0000001) lstY.Items.Add("Деление на 0");

else

lstY.Items.Add((Math.Pow(i, (2.0 / 3.0))) / (i \* i - 1));

break;

case 3:

lstY.Items.Add(Math.Acos(i));

break;

case 4:

lstY.Items.Add(Math.Sqrt(i));

break;

case 5:

lstY.Items.Add(Math.Cos(i));

break;

case 6:

lstY.Items.Add(Math.Asin(i));

break;

case 7:

lstY.Items.Add(Math.Log(i));

break;

case 8:

lstY.Items.Add(Math.Log10(i));

break;

case 9:

lstY.Items.Add(Math.Atan(i));

break;

case 10:

lstY.Items.Add(i \* i);

break;

case 11:

lstY.Items.Add(Math.Tan(i));

break;

}

arg = arg + setstep;

string x = lstX.Items[counter].ToString();

string y = lstY.Items[counter].ToString();

sw.Write(x);

sw.Write("=");

sw.Write(y);

sw.Write("\n");

counter++;

}

sw.Close();

}

}

}

private void lstY\_SelectedIndexChanged(object sender, EventArgs e)

{

lstX.SelectedIndex = lstY.SelectedIndex;

}

private void lstY1\_SelectedIndexChanged(object sender, EventArgs e)

{

lstX1.SelectedIndex = lstY1.SelectedIndex;

}

private void lstX\_SelectedIndexChanged(object sender, EventArgs e)

{

lstY.SelectedIndex = lstX.SelectedIndex;

}

private void lstX1\_SelectedIndexChanged(object sender, EventArgs e)

{

lstY1.SelectedIndex = lstX1.SelectedIndex;

}

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

{

string round(string rounding)

{

double temp = Double.Parse(rounding);

temp = Math.Round(temp, 1);

rounding = temp.ToString();

return rounding;

}

lstX1.Items.Clear();

lstY1.Items.Clear();

lstX1.Items.Add("Аргументы");

lstY1.Items.Add("Значения функции");

lstX1.Visible = true;

lstY1.Visible = true;

string text = File.ReadAllText("Data.txt");

char[] check = { '$', '=' };

string xy = "$";

foreach (var ch in text)

{

string temp = "";

temp += ch;

if (xy.Substring(xy.Length - 1, 1).Equals("="))

{

xy = xy.Trim(check);

lstX1.Items.Add(xy);

xy = "$";

}

if (temp.Equals("\n"))

{

xy = xy.Trim(check);

if (xy.IndexOf(",") > -1)

{

xy = round(xy);

}

lstY1.Items.Add(xy);

xy = "$";

}

xy += ch;

}

}

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

{

foreach (Control control in Controls)

{

control.Visible = true;

}

lstX1.Visible = false;

lstY1.Visible = false;

}

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

{

ChildForm Child = new ChildForm();

Child.Owner = this;

Child.Text = "Справка";

Child.ShowDialog();

Child.Dispose();

}

}

public partial class ChildForm : Form

{

private TextBox msg;

public ChildForm()

{

msg = new TextBox();

msg.Location = new Point(0, 0);

msg.Size = new Size(this.Width, this.Height);

msg.Multiline = true;

msg.Font = new Font("Times New Roman", 18);

msg.ReadOnly = true;

msg.Text = "Данная программа реализует автоматическое построение значений абсцисы и ординаты функции\n" +

"на промежутке между двумя задаваемыми значениями с шагом 1. Пожалуйста, введите значение\n" +

"нижней границы промежутка в левое заполняемое поле, а значение верхней границы промежутка\n" +

"в правое, после чего выберите одну из доступных функций и нажмите на кнопку (Нажмите, чтобы\n" +

"начать работу). В результате левый столбец будет заполнен значениями абсциссы, а правый\n" +

"значениями ординаты. Результаты автоматически выгружаются в файл Data.txt откуда их можно\n" +

"выгрузить обратно в программу с округлением по нажатию кнопки выгрузить. Результаты будут в отдельной таблице\n";

Controls.Add(msg);

Resize += msg\_Resize;

}

private void msg\_Resize(object sender, EventArgs e)

{

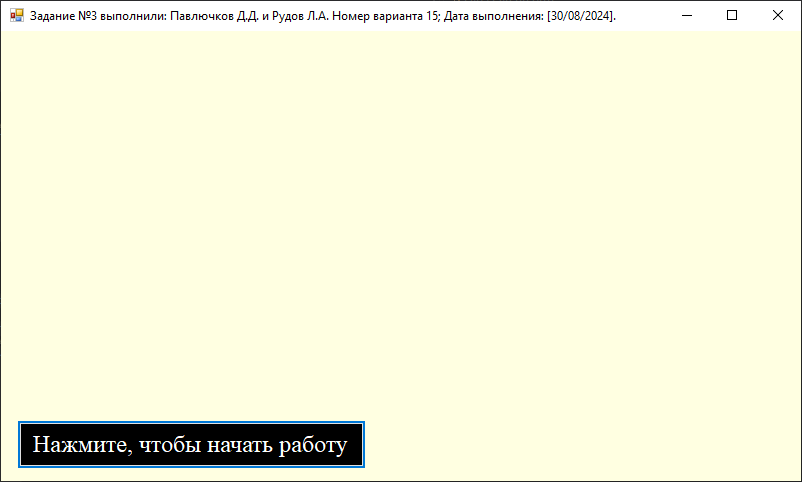
msg.Size = new Size(this.Width, this.Height);

}

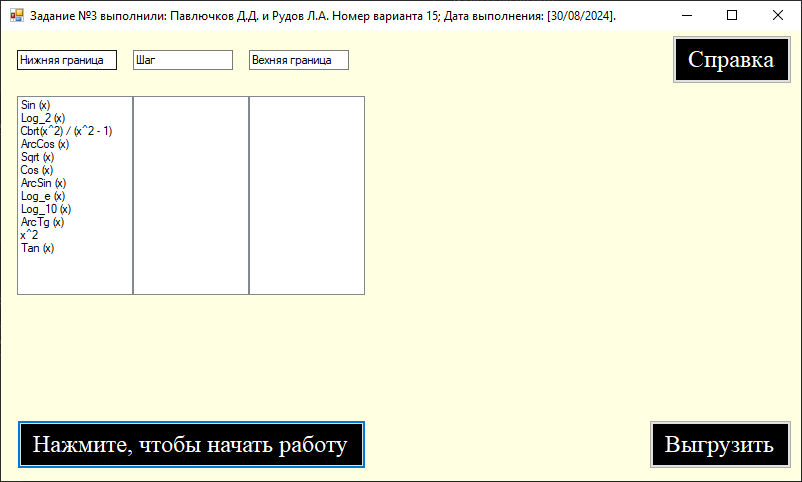
}

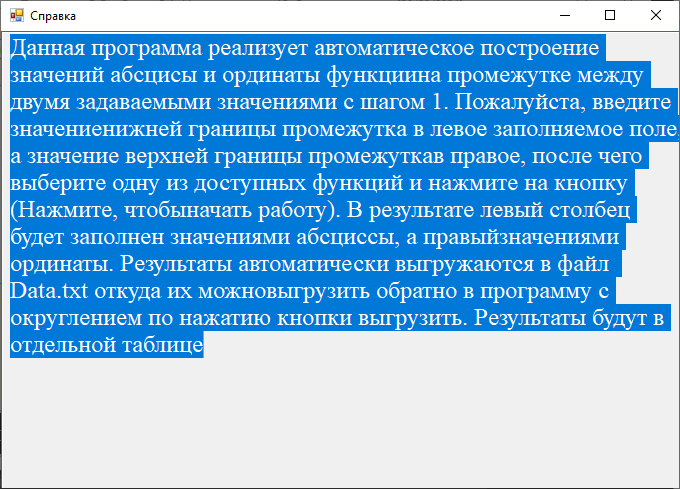
}

**Графический пользовательский интерфейс**

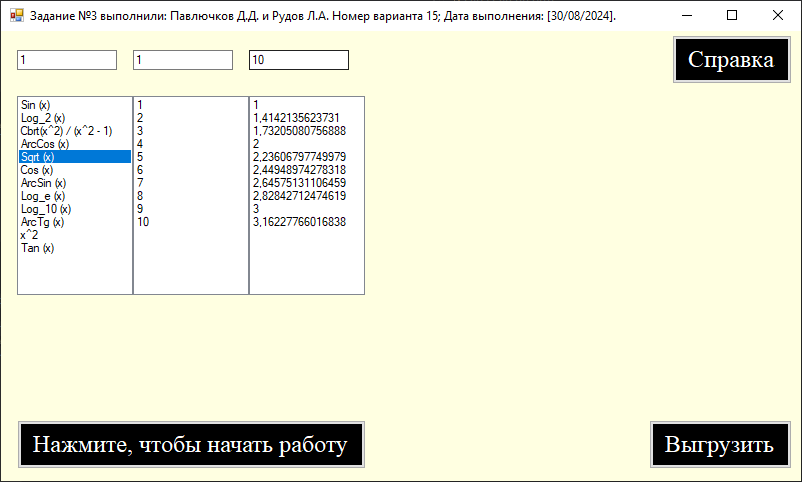
****

ГПИ стартовой страницы

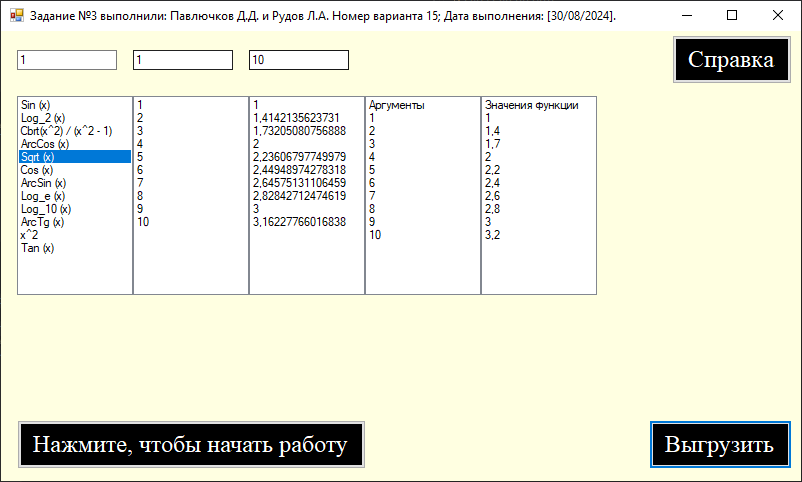
****

ГПИ основной страницы после начала работы с программой****

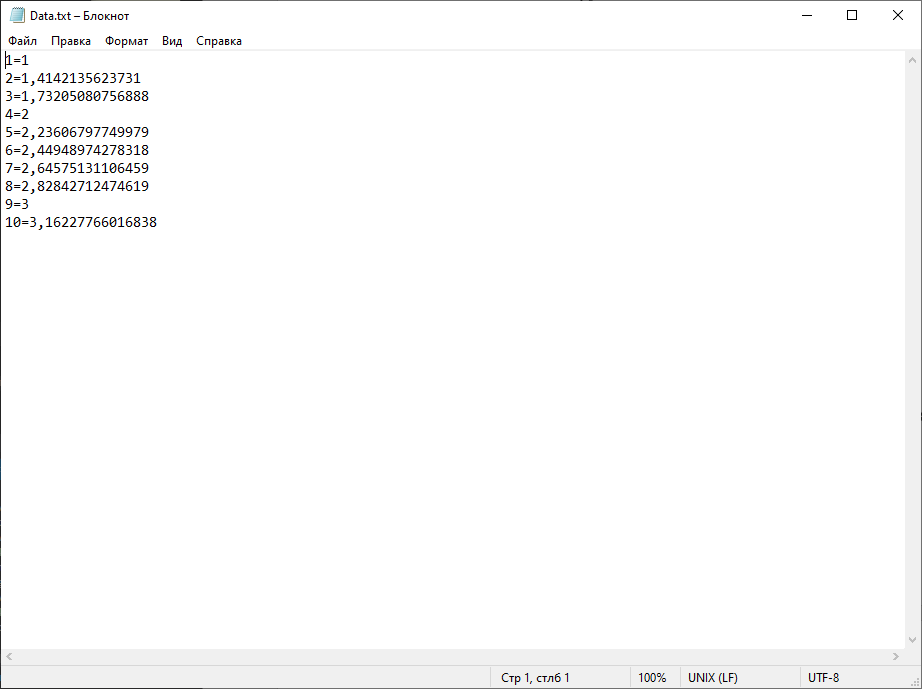
Справка, вызываемая нажатием на кнопку «Справка»



Результат работы программы с указанными функцией и границами



Результат работы программы с указанными функцией и границами после нажатия кнопки Выгрузить



Содержимое файла Data.txt, в который производится запись значений.

**7. Вывод**

В ходе выполнения лабораторной работы были закреплены навыки разработки визуального пользовательского интерфейса, освоена работа с текстовыми файлами в среде *Microsoft Visual Studio*, получены навыки работы с элементами управления.