Министерство транспорта Российской Федерации

Федеральное государственное автономное

образовательное учреждение высшего образования

«Российский университет транспорта (МИИТ)»

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

Лабораторная работа №3

по дисциплине:

«Методы программирования»

на тему:  
«Автоматизация компоновки ГПИ. Функции»

Выполнил: ст. гр. ТКИ-341

Черкас Н.В., Родин К.П.

Вариант №12

Проверил: к.т.н., доцент Сафронов А.И.

Москва – 2024 г.

# **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. Формулировка задачи**

«В интегрированной среде разработки *Visual Studio* разработать программу в режиме *Windows Forms Application* на языке *Visual C#*, представляющую собой пользовательский интерфейс, содержащий главное меню, позволяющее:

1. Начать работу с приложением.
2. Прервать работу приложения.
3. Предоставить пользователю справочную информацию о работе с приложением.

Сама программа должна реализовывать вывод в списки значений аргумента и соответствующих им значений функций. Список функций должен обязательно содержать следующие пункты:

1. Логарифм по основанию 2,
2. Тангенс,
3. Возведение в квадрат,
4. Косинус,
5. Натуральный логарифм,
6. Арккосинус,
7. Извлечение корня,
8. Арктангенс,
9. Синус,
10. Десятичный логарифм,
11. Арксинус.

Согласно заданию, список должен быть организован в виде обыкновенного списка. Функции из списка должны быть расположены в следующем порядке: 2, 1, 10, 11, 5, 9, 8, 6, 7, 3, 4. Так же необходимо поместить дополнительную функциональную зависимость ln(x) / x на 4 позицию. Таким образом, функции должны быть расположены в следующем порядке: Тангенс, Логарифм по основанию 2, Десятичный логарифм, ln(x) / x, Арксинус, Натуральный Логарифм, Синус, Арктангенс, Арккосинус, Извлечение корня, Возведение в квадрат, Косинус.

# **3. Составление диаграммы классов, входящих в состав решения.**

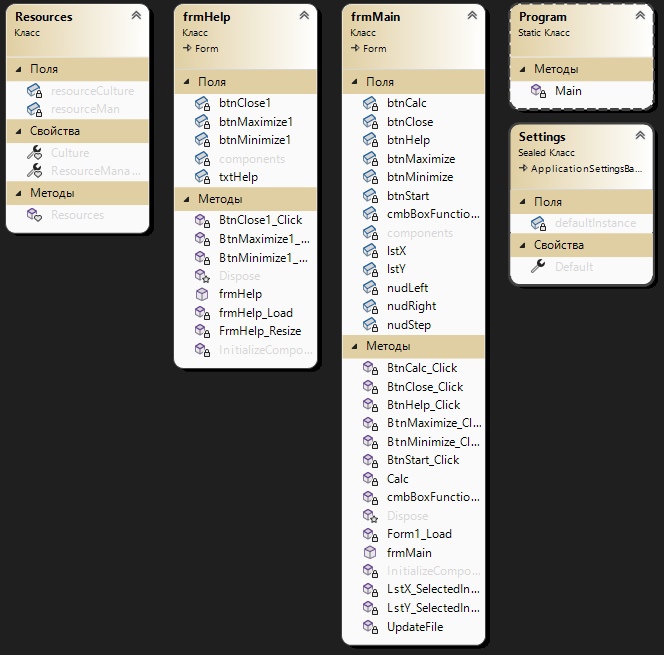


Рисунок 1 – Диаграмма классов

# **4. Составление сети Петри запрограммированного технологического процесса.**

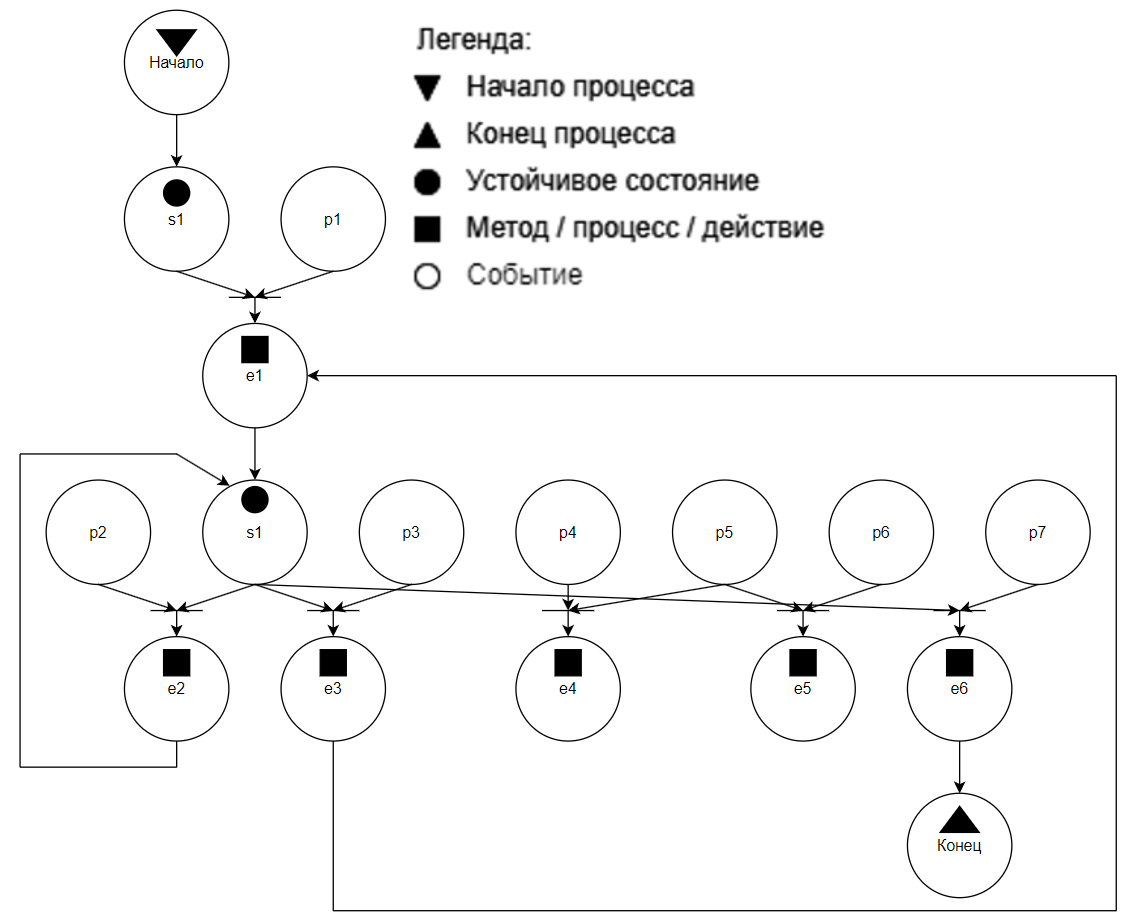


Рисунок 2 - Сеть Петри основной формы

**Описание сети Петри**

- состояния (states)

s1 – форма ожидает действий пользователя

- действия (effects)

e1 – разблокируются кнопки взаимодействия;

кнопка «Начало работы с приложением» деактивируется

e2 – запись в файл аргументов и результатов функции

e3 – вывод результатов из файла в списки

e4 – выбор соответствующего элемента из списка ординат

e5 – выбор соответствующего элемента из списка абсцисс

e6 – закрытие формы

- события (prompts)

p1 – нажата кнопка «Начало работы с приложением»

p2 – в списке выбрана функция

p3 – нажатие кнопки рассчитать

p4 – пользователь выбрал элемент из списка абсцисс

p5 – списки заполнены значениями

p6 – пользователь выбрал элемент из списка ординат

p7 – нажата кнопка закрытия

# **5. Составление схем алгоритмов методов в составе решения, отмеченных на сети Петри в качестве «эффектов» (метка ).**



Рисунок 3 – Алгоритм кнопки "Начало работы с приложением"

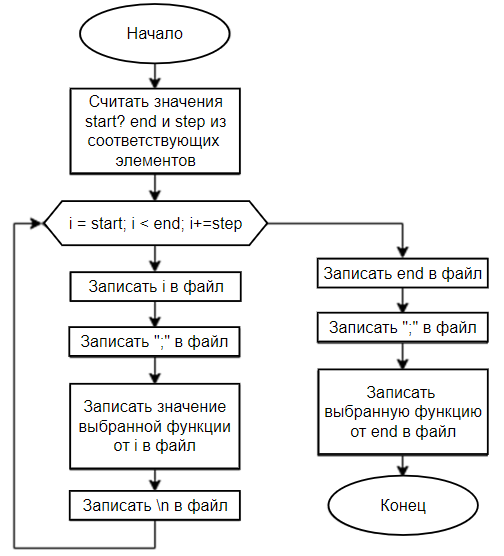


Рисунок 4 - Алгоритм записи информации в файл

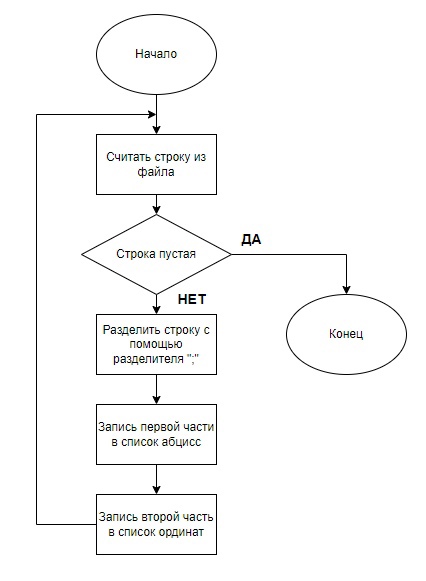


Рисунок 5 - Алгоритм чтения информации из файла

# **6. Подбор тестовых примеров.**

Тестовые примеры:

* Проверить корректность вычисления каждой из формул
* Проверить запись в файл

**7. Листинг (код) составленного программного обеспечения.**

Основная форма:

using System;

using System.Drawing;

using System.IO;

using System.Windows.Forms;

namespace Task3

{

public partial class frmMain : Form

{

private ListBox lstX;

private ListBox lstY;

private NumericUpDown nudLeft;

private NumericUpDown nudRight;

private NumericUpDown nudStep;

private Button btnCalc;

private Button btnStart;

private Button btnClose;

private Button btnMinimize;

private Button btnMaximize;

private Button btnHelp;

private ComboBox cmbBoxFunctions;

public frmMain()

{

InitializeComponent();

this.ControlBox = false;

}

private void Form1\_Load(object sender, EventArgs e)

{

this.Text = "Задание №3 выполнили: Черкаc НВ, Родин КП; Номер варианта: 12; Дата выполнения: 11/04/2024";

this.BackColor = SystemColors.Control;

this.ClientSize = new Size(400, 310);

this.FormBorderStyle = FormBorderStyle.FixedSingle;

this.lstX = new ListBox();

this.lstY = new ListBox();

this.nudLeft = new NumericUpDown();

this.nudRight = new NumericUpDown();

this.nudStep = new NumericUpDown();

this.btnCalc = new Button();

this.btnStart = new Button();

this.btnClose = new Button();

this.btnMinimize = new Button();

this.btnMaximize = new Button();

this.btnHelp = new Button();

this.cmbBoxFunctions = new ComboBox();

//cmbBoxFunctions

this.cmbBoxFunctions.Anchor = ((System.Windows.Forms.AnchorStyles.Bottom | System.Windows.Forms.AnchorStyles.Left)

| System.Windows.Forms.AnchorStyles.Right);

this.cmbBoxFunctions.DropDownWidth = 160;

this.cmbBoxFunctions.Items.AddRange(new object[] { "Тангенс", "Логарифм по основанию 2", "Десятичный логарифм", "ln(x)/x", "Арксинус", "Натуральный логарифм", "Синус", "Арктангенс", "Арккосинус", "Извлечение корня", "Возведение в квадрат", "Косинус" });

this.cmbBoxFunctions.Location = new System.Drawing.Point(230, 10);

this.cmbBoxFunctions.Size = new System.Drawing.Size(160, 100);

this.cmbBoxFunctions.TabIndex = 7;

this.cmbBoxFunctions.DropDownStyle = System.Windows.Forms.ComboBoxStyle.DropDownList;

//listX

this.lstX.Location = new Point(10, 10);

this.lstX.Size = new Size(100, 300);

this.lstX.SelectedIndexChanged += LstX\_SelectedIndexChanged;

//listY

this.lstY.Location = new Point(120, 10);

this.lstY.Size = new Size(100, 300);

this.lstY.SelectedIndexChanged += LstY\_SelectedIndexChanged;

//Выбор левой границы

this.nudLeft.Location = new Point(230, 40);

this.nudLeft.Size = new Size(50, 20);

this.nudLeft.Minimum = -100;

this.nudLeft.Maximum = 100;

this.nudLeft.Value = -5;

this.nudLeft.DecimalPlaces = 1;

this.nudLeft.Increment = (decimal)0.1;

//Выбор правой границы

this.nudRight.Location = new Point(285, 40);

this.nudRight.Size = new Size(50, 20);

this.nudRight.Minimum = -100;

this.nudRight.Maximum = 100;

this.nudRight.Value = 5;

this.nudRight.DecimalPlaces = 1;

this.nudRight.Increment = (decimal)0.1;

//Выбор шага

this.nudStep.Location = new Point(340, 40);

this.nudStep.Size = new Size(50, 20);

this.nudStep.Minimum = (decimal)0.1;

this.nudStep.Maximum = 10;

this.nudStep.Value = (decimal)0.1;

this.nudStep.DecimalPlaces = 1;

this.nudStep.Increment = (decimal)0.1;

//Кнопка "Вычислить"

this.btnCalc.Location = new Point(230, 70);

this.btnCalc.Size = new Size(160, 80);

this.btnCalc.FlatStyle = FlatStyle.Flat;

this.btnCalc.FlatAppearance.BorderSize = 1;

this.btnCalc.Text = "Вычислить";

this.btnCalc.Click += BtnCalc\_Click;

//Кнопка "Старт"

this.btnStart.Location = new Point(150, 135);

this.btnStart.Size = new Size(100, 40);

this.btnStart.FlatStyle = FlatStyle.Flat;

this.btnStart.FlatAppearance.BorderSize = 1;

this.btnStart.Text = "Старт";

this.btnStart.Click += BtnStart\_Click;

//Кнопка "Справка"

this.btnHelp.Location = new Point(230, 160);

this.btnHelp.Size = new Size(160, 80);

this.btnHelp.FlatStyle = FlatStyle.Flat;

this.btnHelp.FlatAppearance.BorderSize = 1;

this.btnHelp.Text = "Справка";

this.btnHelp.Click += BtnHelp\_Click;

//Кнопка "Закрыть"

this.btnClose.Location = new Point(370,280);

this.btnClose.Size = new Size(20, 20);

this.btnClose.FlatStyle = FlatStyle.Flat;

this.btnClose.FlatAppearance.BorderSize = 1;

this.btnClose.Text = "X";

this.btnClose.Click += BtnClose\_Click;

//Кнопка "Минимизировать"

this.btnMinimize.Location = new Point(320, 280);

this.btnMinimize.Size = new Size(20, 20);

this.btnMinimize.FlatStyle = FlatStyle.Flat;

this.btnMinimize.FlatAppearance.BorderSize = 1;

this.btnMinimize.Text = "-";

this.btnMinimize.Click += BtnMinimize\_Click;

//Кнопка "Максимизировать"

this.btnMaximize.Location = new Point(345, 280);

this.btnMaximize.Size = new Size(20, 20);

this.btnMaximize.FlatStyle = FlatStyle.Flat;

this.btnMaximize.FlatAppearance.BorderSize = 1;

this.btnMaximize.Text = "⬜";

this.btnMaximize.Click += BtnMaximize\_Click;

this.Controls.AddRange(new Control[] {cmbBoxFunctions, lstX, lstY, nudLeft, nudRight, nudStep, btnCalc, btnClose, btnMinimize, btnMaximize, btnHelp });

foreach (Control control in this.Controls)

{

control.Enabled = false;

}

this.Controls.Add(btnStart);

foreach (Control control in this.Controls)

{

control.BackColor = SystemColors.Control;

control.ForeColor = SystemColors.ControlText;

}

this.btnStart.BringToFront();

}

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

{

foreach (Control control in this.Controls)

{

control.Visible = true;

control.Enabled = true;

}

this.btnStart.Visible = false;

}

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

{

frmHelp fr2 = new frmHelp();

fr2.Show();

Hide();

}

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

{

Application.Exit();

}

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

{

this.WindowState = FormWindowState.Minimized;

}

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

{

if (this.WindowState == FormWindowState.Normal)

{

this.WindowState = FormWindowState.Maximized;

}

else

{

this.WindowState = FormWindowState.Normal;

}

}

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

{

this.lstX.SelectedIndex = this.lstY.SelectedIndex;

}

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

{

this.lstY.SelectedIndex = this.lstX.SelectedIndex;

}

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

{

this.UpdateFile();

this.lstX.Items.Clear();

this.lstY.Items.Clear();

StreamReader sr = new StreamReader("C:\\Users\\Никита\\Desktop\\Миит\\Методы программирования Сафронов\\Задание3\\FunctionArguments.txt");

while (true)

{

string str = sr.ReadLine();

if (str == null) break;

string[] strarr = str.Split(';');

this.lstX.Items.Add(strarr[0]);

this.lstY.Items.Add(strarr[1]);

}

sr.Close();

}

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

{

this.UpdateFile();

}

private void UpdateFile()

{

StreamWriter sw = new StreamWriter("C:\\Users\\Никита\\Desktop\\Миит\\Методы программирования Сафронов\\Задание3\\FunctionArguments.txt");

decimal start = this.nudLeft.Value;

decimal end = this.nudRight.Value;

decimal step = this.nudStep.Value;

for (decimal i = start; i < end; i += step)

{

sw.Write(i);

sw.Write(";");

sw.Write(this.Calc(i));

sw.Write("\n");

}

sw.Write(end);

sw.Write(";");

sw.Write(this.Calc(end));

sw.Close();

}

private string Calc(decimal arg)

{

switch (this.cmbBoxFunctions.SelectedIndex)

{

case 0:

if (Math.Cos((double)arg) != 0) return Math.Tan((double)arg).ToString("F3");

else return "ERROR";//2

case 1:

if (arg > 0) return Math.Log((double)arg, 2).ToString("F3");

else return "ERROR";//1

case 2:

if (arg > 0) return Math.Log((double)arg, 10).ToString("F3");

else return "ERROR";//10

case 3:

if ((arg != 0) && (arg>0)) return (Math.Log((double)arg, Math.E) / (double)arg).ToString("F3");

else return "ERROR";//c

case 4:

if (Math.Abs(arg) > 1) return "ERROR";

else return Math.Asin((double)arg).ToString("F3");//11

case 5:

if (arg > 0) return Math.Log((double)arg, Math.E).ToString("F3");

else return "ERROR";//5

case 6:

return Math.Sin((double)arg).ToString("F3");//9

case 7:

return Math.Atan((double)arg).ToString("F3");//8

case 8:

if (Math.Abs(arg) > 1) return "ERROR";

else return Math.Acos((double)arg).ToString("F3");//6

case 9:

if (arg >= 0) return Math.Sqrt((double)arg).ToString("F3");

else return "ERROR";//7

case 10:

return Math.Pow((double)arg, 2).ToString("F3");//3

default:

return Math.Cos((double)arg).ToString("F3");//4

}

}

}

}

Дополнительная форма:

using System;

using System.Drawing;

using System.IO;

using System.Windows.Forms;

namespace Task3

{

public partial class frmHelp : Form

{

private Label txtHelp;

private Button btnClose1;

private Button btnMinimize1;

private Button btnMaximize1;

public frmHelp()

{

this.ControlBox = false;

this.txtHelp = new Label();

this.SuspendLayout();

//

// txtHelp

//

this.txtHelp.Location = new System.Drawing.Point(0, 0);

this.txtHelp.Size = new System.Drawing.Size(700,150);

//

// frmHelp

//

this.ClientSize = new System.Drawing.Size(700, 400);

this.Controls.Add(this.txtHelp);

this.Name = "frmHelp";

this.Text = "Справка";

this.Load += new System.EventHandler(this.frmHelp\_Load);

this.ResumeLayout(false);

this.btnClose1 = new Button();

this.btnMinimize1 = new Button();

this.btnMaximize1 = new Button();

//Кнопка "Закрыть"

this.btnClose1.Location = new Point(60, 150);

this.btnClose1.Size = new Size(20, 20);

this.btnClose1.FlatStyle = FlatStyle.Flat;

this.btnClose1.FlatAppearance.BorderSize = 1;

this.btnClose1.Text = "X";

this.btnClose1.Click += BtnClose1\_Click;

//Кнопка "Минимизировать"

this.btnMinimize1.Location = new Point(10, 150);

this.btnMinimize1.Size = new Size(20, 20);

this.btnMinimize1.FlatStyle = FlatStyle.Flat;

this.btnMinimize1.FlatAppearance.BorderSize = 1;

this.btnMinimize1.Text = "-";

this.btnMinimize1.Click += BtnMinimize1\_Click;

//Кнопка "Максимизировать"

this.btnMaximize1.Location = new Point(35, 150);

this.btnMaximize1.Size = new Size(20, 20);

this.btnMaximize1.FlatStyle = FlatStyle.Flat;

this.btnMaximize1.FlatAppearance.BorderSize = 1;

this.btnMaximize1.Text = "⬜";

this.btnMaximize1.Click += BtnMaximize1\_Click;

this.Controls.AddRange(new Control[] {btnClose1, btnMinimize1, btnMaximize1 });

foreach (Control control in this.Controls)

{

control.BackColor = SystemColors.Control;

control.ForeColor = SystemColors.ControlText;

}

this.btnClose1.BringToFront();

this.btnMinimize1.BringToFront();

this.btnMaximize1.BringToFront();

}

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

{

StreamReader sr = new StreamReader("C:\\Users\\Никита\\Desktop\\Миит\\Методы программирования Сафронов\\Задание3\\Help.txt");

txtHelp.Text = sr.ReadToEnd();

sr.Close();

this.Resize += FrmHelp\_Resize;

}

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

{

this.txtHelp.Size = this.ClientSize;

}

private void BtnClose1\_Click(object sender, EventArgs e)

{

frmMain fr1 = new frmMain();

fr1.Show();

Hide();

}

private void BtnMinimize1\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void BtnMaximize1\_Click(object sender, EventArgs e)

{

if (this.WindowState == FormWindowState.Normal)

{

this.WindowState = FormWindowState.Maximized;

}

else

{

this.WindowState = FormWindowState.Normal;

}

}

}

}

# **8. Графический пользовательский интерфейс программного обеспечения и его описание.**

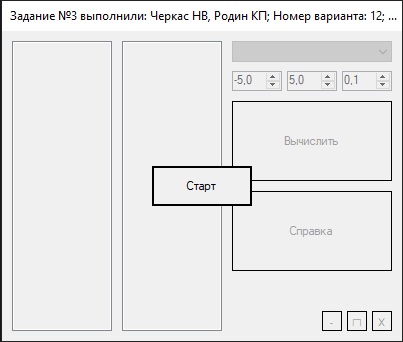


Рисунок 6 - Интерфейс основной формы до нажатия кнопки «Начало работы с приложением»

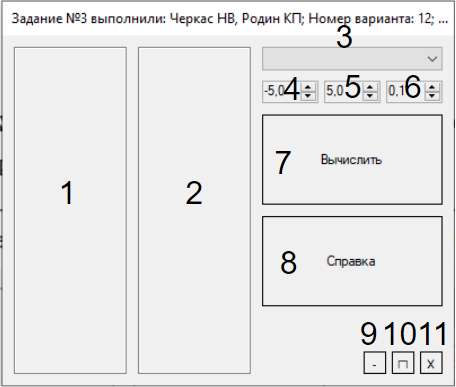


Рисунок 7 – Интерфейс основной формы после нажатия кнопки «Начало работы с приложением

1. Список абсцисс
2. Список ординат
3. Комбинированный список с функциями
4. Поле для выбора левой границы интервала
5. Поле для выбора правой границы интервала
6. Поле для выбора значения шага функции
7. Кнопка расчета
8. Кнопка вызова формы со справкой
9. Кнопка сворачивания функции
10. Кнопка максимизации/минимизации функции
11. Кнопка закрытия формы

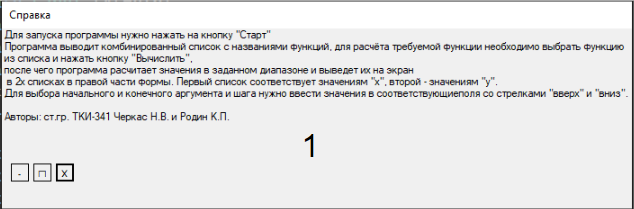


Рисунок 8 – Интерфейс вспомогательной формы

1. Зона для выгрузки текста из файла

# **9. Подтверждение соответствия графического пользовательского интерфейса требованиям к оформлению.**

1. Заголовок экранной формы должен содержать надпись вида: «Задание №3 выполнил: [Фамилия И.О. автора]; Номер варианта: [Номер]; Дата выполнения: [дд/мм/гггг].



Рисунок 9 - Название формы

2. Дата выполнения проставляется в момент, когда программа считается законченной и по ней можно готовить итоговый отчёт о выполнении работы.



Рисунок 10 - Дата на момент написания отчета

3. Чётные варианты отключают стандартный блок управления экранной формой и создают авторские кнопки «Свернуть», «Развернуть», «Закрыть» внизу экранной формы.



Рисунок 11 - Авторские кнопки «Свернуть», «Развернуть», «Закрыть»

5. Справочная информация должна быть вызвана в дочерней экранной форме и считана в статический по размеру ярлык из текстового файла (чётные варианты).

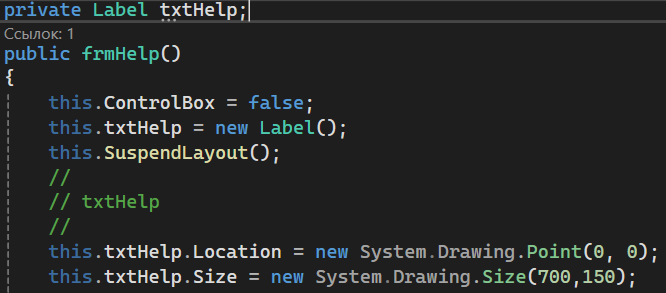


Рисунок 12 - Фрагмент кода с созданием ярлыка для выгрузки справочной информации

6. В режиме конструктора экранная форма не должна содержать ни одного элемента управления.

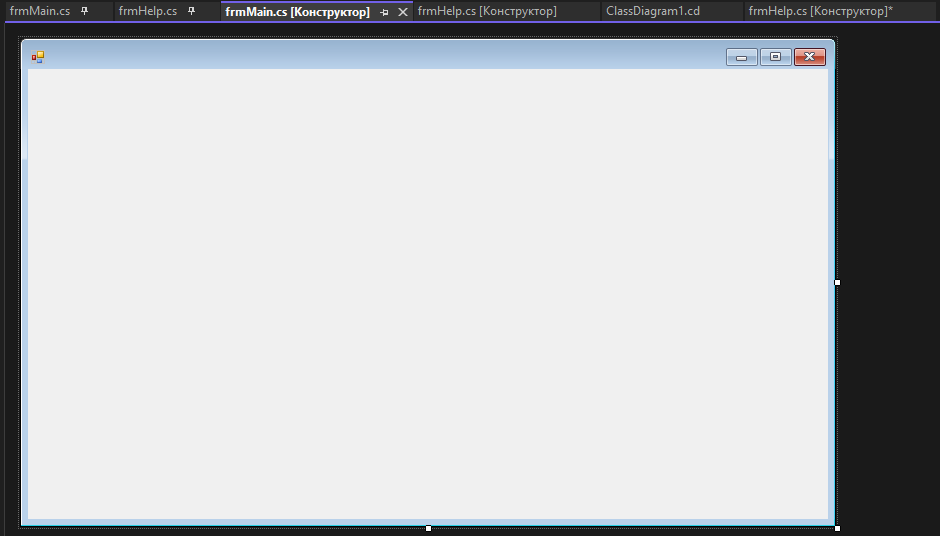


Рисунок 13 - Конструктор главной формы

7. Решение (*solution*) должно содержать только одну единственную экранную форму. Экранная форма со справкой настраивается программным кодом.

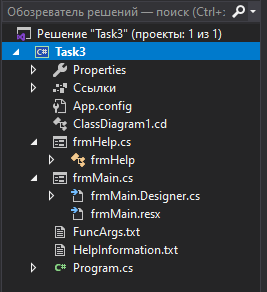


Рисунок 14 - Решение (solution)

8. Компоненты экранной формы со справкой должны быть чувствительными к изменению размера экранной формы по ширине и высоте (обязательно подключение события *Resize*).

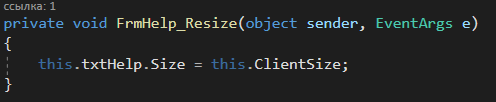


Рисунок 15 - Фрагмент кода с изменением размера компонента экранной формы со справкой

9. Названия функций должны быть элементами комбинированного списка (чётные варианты).

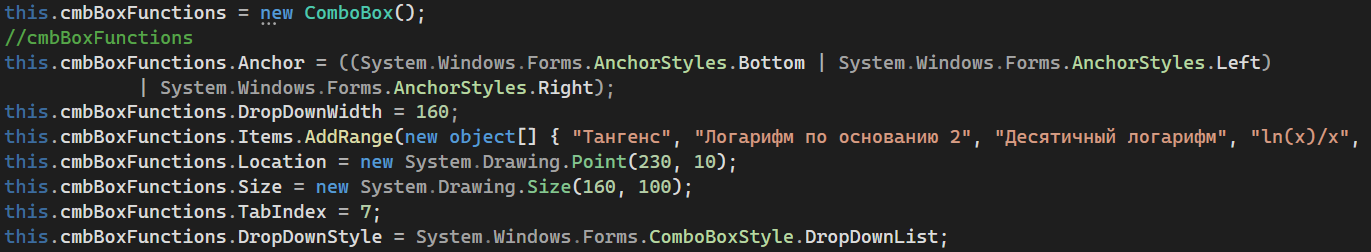


Рисунок 16 - Фрагмент кода с созданием комбинированного списка с функциями

10. Выбор конкретной функции из списка инициирует заполнение некоторого файла значениями аргумента и значениями функции. Разделитель данных подбирается согласно таблице вариантов.

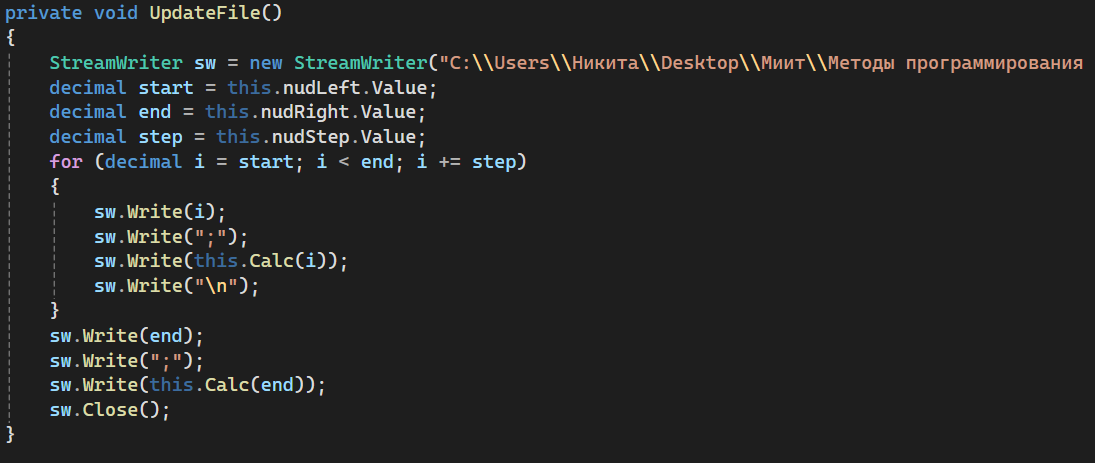


Рисунок 17 - Фрагмент кода с функцией заполнения файла

Таблица 2 – Варианты разделителей, записываемых в файл

|  |  |
| --- | --- |
| 1. | $ |
| 2. | % |
| 3. | &\* |
| 4. | <> |
| 5. | { |
| 6. | $#@ |
| 7. | ^^ |
| 8. | ( |
| 9. | . |
| 10. | >> |
| 11. | , |
| 12. | ; |
| 13. | : |
| 14. | ++ |
| 15. | = |
| 16. | !@ |
| 17. | ||| |
| 18. | пробел |
| 19. | \_ |
| 20. | … |

12. Предусмотреть кнопку, нажатие на которую инициирует заполнение двух списков (один список под значения аргумента, другой список – под значения функции, округлённых по правилу *(N mod 5) + 1* для знаков после плавающей запятой, где *N* – номер варианта) данными о функциональной зависимости из файла. Обучающиеся с чётными номерами вариантов обновляют существующие основные списки, с нечётными номерами вариантов выводят значения, считанные из файла, в дополнительные списки.

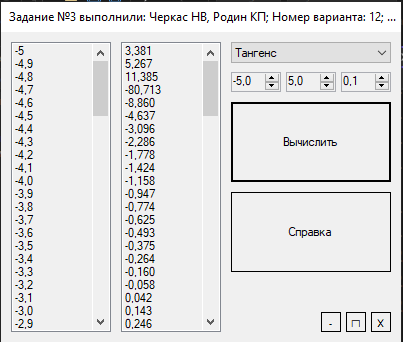


Рисунок 18 - Результат работы функции расчета

13. В качестве исходного состояния функциональных компонентов главной экранной формы принимается либо недоступное состояние (*.Enabled = false*) – чётные варианты.

см. Рисунок 6

14. Кнопка «Начало работы с приложением» должна реализовывать активацию доступа пользователя к функциональным компонентам или отображение функциональных компонентов на экранной форме для пользователя.

см. Рисунок 7

15. Все элементы программы должны носить значащие имена переменных, в которых отражено существо этих элементов, например, экранная форма – *frmMain*, ярлык – *lblHelp*, комбинированный список – *cmbFigures* и т.д.

см. Диаграмму классов

16. Выбор значения абсциссы в списке должен инициировать выбор соответствующего ему значения ординаты и наоборот – выбор значения ординаты должен инициировать выбор соответствующего ему значения абсциссы (работа с параметрами *SelectedIndex* или *SelectedItem* списков).

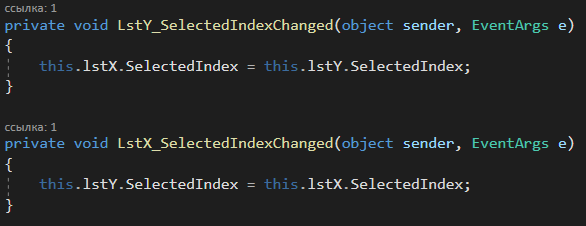


Рисунок 19 - Фрагмент кода с функциями обработки выбора значения в списках абсцисс или ординат

# **10. Расчёт тестовых примеров с использованием составленного программного обеспечения.**

* Проверить корректность вычисления каждой из формул

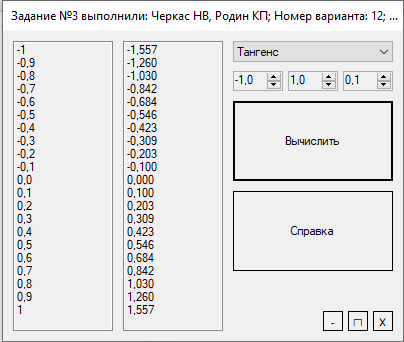


Рисунок 20 - Вычисление функции Тангенс

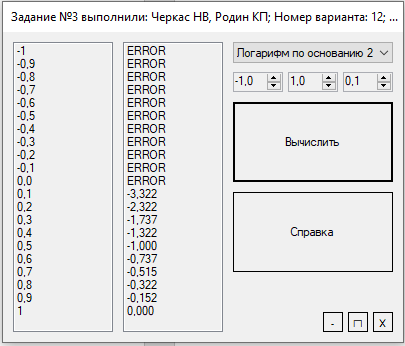


Рисунок 21 - Вычисление функции Логарифм по основанию 2

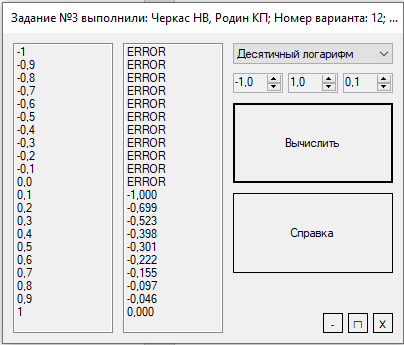


Рисунок 22 - Вычисление функции Десятичный логариф

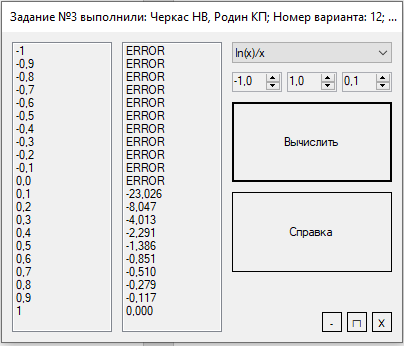


Рисунок 23 - Вычисление функции ln(x) / x

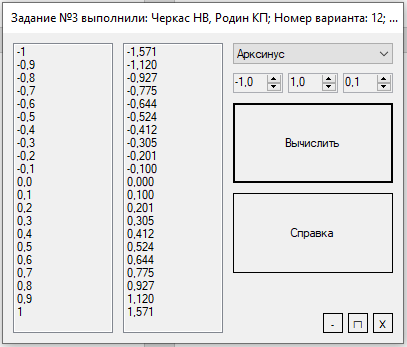


Рисунок 24 - Вычисление функции Арксинус

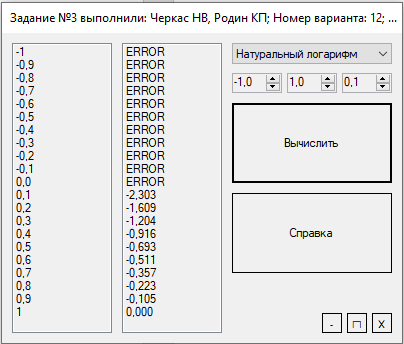


Рисунок 25 - Вычисление функции Натуральный логарифм

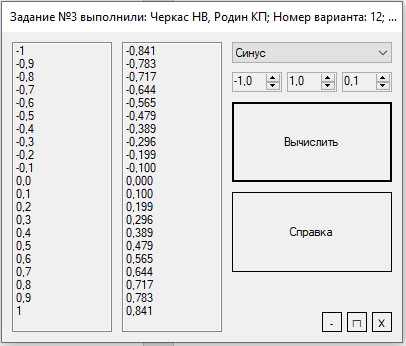


Рисунок 26 - Вычисление функции Синус

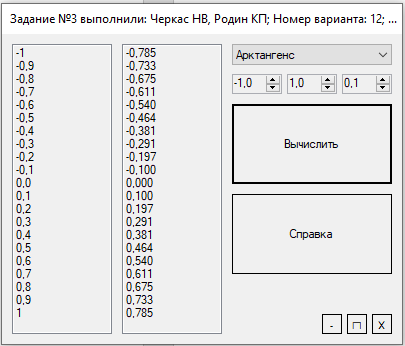


Рисунок 27 - Вычисление функции Арктангенс

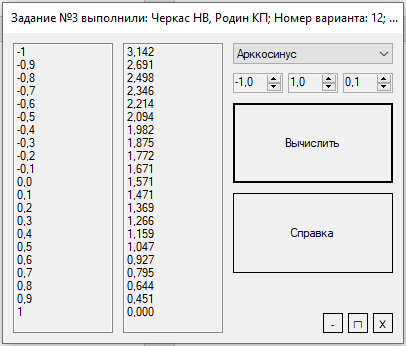


Рисунок 28 - Вычисление функции Арккосинус

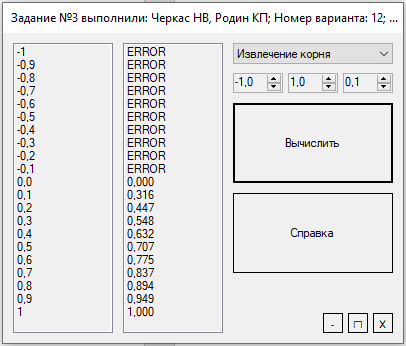


Рисунок 29 - Вычисление функции Извлечение корня

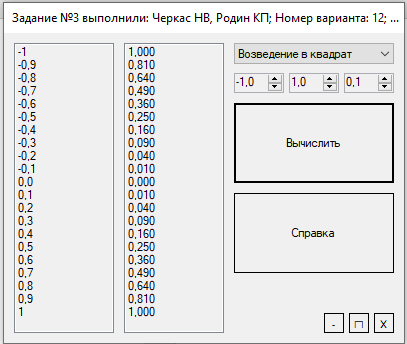


Рисунок 30 - Вычисление функции Возведение в квадрат

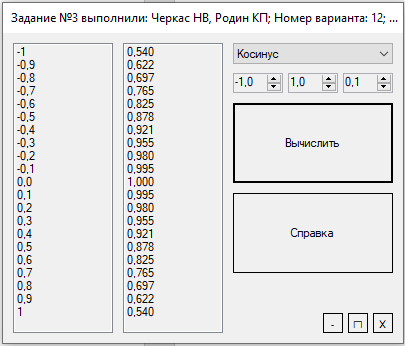


Рисунок 31 - Вычисление функции Косинус

* Проверить запись в файл

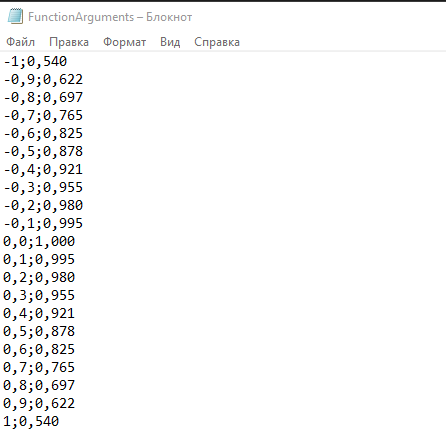


Рисунок 32 - Записи в файле

# **11. Формулировка вывода о проделанной работе**

В рамках данной работы были закреплены навыки разработки визуального пользовательского интерфейса, настраиваемого программным кодом без использования «Конструктора», освоена работа с текстовыми файлами и кодировкой в среде *Microsoft* *Visual Studio С#*, были получены навыки реализации программного обеспечения с использованием взаимного увязывания одних элементов управления с другими, а также получено представление о перерисовке и перемасштабировании.