**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Санкт-Петербургский государственный технологический институт**

**(технический университет)»**

**(СПбГТИ (ТУ))**

**Отчет об учебной практике**

**Тема:** *«***Разработка программного комплекса для поддержки решения при анализе экспериментальных данных на примере исследования параболической функции»**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| Направление подготовки | *09.03.01* | *Информатика и вычислительная техника* |
| Факультет | *Информационных технологий и управления* | |
| Кафедра | *Систем автоматизированного проектирования и управления* | |
| Группа | *474* | |
|  |  | |

*Студент Кузьмин В.Г.*

*(подпись, дата) (инициалы, фамилия)*

*Руководитель, Уланов В.Н.*

*доц., к.т.н. (подпись, дата) (инициалы, фамилия)*

*Оценка \_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

*(подпись руководителя)*

Санкт-Петербург

2019

**Оглавление**

[1. Цель учебной практики 3](#_Toc13642613)

[2.Задание на учебную практику 3](#_Toc13642614)

[3. План выполнения задания 3](#_Toc13642615)

[3.1. Знакомство с объектом приложения 3](#_Toc13642616)

[4. Выбор инструментария 4](#_Toc13642617)

[5. Выбор языка программирования 4](#_Toc13642618)

[6.Анализ характеристик альтернативных программных комплексов 5](#_Toc13642619)

[7. Интерфейс программы 7](#_Toc13642620)

[8. План тестирования программного комплекса 7](#_Toc13642621)

[9. Тестирование программного комплекса 8](#_Toc13642622)

[9. Анализ выполнения учебной практики 10](#_Toc13642623)

[10. Вывод по учебной практике 10](#_Toc13642624)

[11. Заключение 10](#_Toc13642625)

[ПРИЛОЖЕНИЕ А 11](#_Toc13642626)

# **1. Цель учебной практики**

Целью прохождения учебной практики является реализация полученных теоретических знаний, умений и навыков, а также получение представления о практической деятельности.

# **2.Задание на учебную практику**

В данной работе необходимо создать приложение, реализующее восстановление формулы функции параболы по заданным значениям аргумента, функции и погрешности.

# **3.Дневник выполнения задания**

В соответствии с планом работы, представленным в техническом задании, я начал работу с разработки интерфейса программного продукта, для наилучшего представления о том, что должно получиться в конце. В процессе разработки я принял решение визуально отделить графическое представление данных эксперимента от непосредственного ввода и таблицы, в которой хранятся числовые значения результатов экспериментов. В конечном итоге интерфейс получился интуитивно понятным для пользователя, что для современного рынка важный критерий. В дальнейшем разработка алгоритмов не вызывала затруднений, т.к. я чётко понимал конечную цель и имел необходимую математическую базу.

## **3.1. Знакомство с объектом приложения**

Обозначенная задача восстановления функции по заданным значениям функции и аргумента, решается методом Гаусса. Метод Гаусса – наиболее мощный и универсальный инструмент для нахождения решения любой системы линейных уравнений и заключается в последовательном исключении неизвестных. На вход необходимо подать *границы функции*, на основании которых будут автоматически заполнены аргументы функции, *погрешность*, с помощью которой будут определены интервалы рандомных значений функции. Также, пользователь должен ввести *расчётные коэффициенты* функции с которыми и будут сравниваться *восстановленные коэффициенты*.

**3.2 Блок схема**

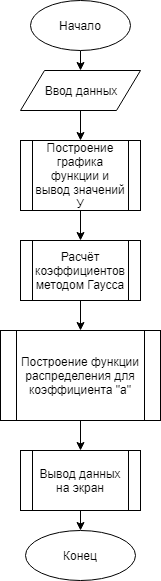


Рис.1 – Блок-схема работы программы

Данная схема отражает этапы работы программы, более подробно один из этапов описан ниже.

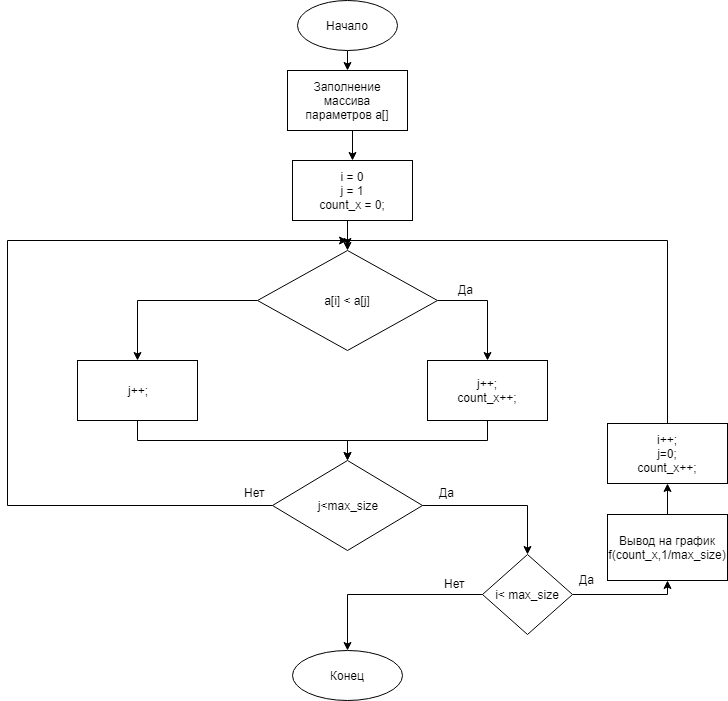


Рис.2 Блок-схема алгоритма построения графика функции распределения

# **4. Выбор инструментария**

Для решения поставленной задачи было решено использовать основных средства Visual Studio. Рассмотрим подробнее.

**Microsoft Visual Studio** — линейка продуктов компании Microsoft, включающих интегрированную среду разработки программного обеспечения и ряд других инструментальных средств. Данные продукты позволяют разрабатывать как консольные приложения, так и приложения с графическим интерфейсом, в том числе с поддержкой технологии Windows Forms, а также веб-сайты, веб-приложения, веб-службы как в родном, так и в управляемом кодах для всех платформ, поддерживаемых Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework и Silverlight.

Visual Studio включает в себя редактор исходного кода с поддержкой технологии IntelliSense и возможностью простейшего рефакторинга кода. Встроенный отладчик может работать как отладчик уровня исходного кода, так и отладчик машинного уровня. Остальные встраиваемые инструменты включают в себя редактор форм для упрощения создания графического интерфейса приложения, веб-редактор, дизайнер классов и дизайнер схемы базы данных. Visual Studio позволяет создавать и подключать сторонние дополнения (плагины) для расширения функциональности практически на каждом уровне, включая добавление поддержки систем контроля версий исходного кода (как, например, Subversion и Visual SourceSafe), добавление новых наборов инструментов (например, для редактирования и визуального проектирования кода на предметно-ориентированных языках программирования) или инструментов для прочих аспектов процесса разработки программного обеспечения (например, клиент Team Explorer для работы с Team Foundation Server).

# **5. Выбор языка программирования**

**Сходства:**  
в С# сохранены основные возможности языка C++, которые не были включены в Java. Это, касается, например, перечислений. Более того в C# для перечислений обеспечивается безопасность типов. С# заимствовал из С++ понятия операторов и ключевых слов. Обеспечивается перегрузка операторов. В C# реализована возможность добавления блоков ненадежного кода.

**Различия:**

В C# использован комплексный метод программирования. При C++ про граммерам приходилось объявлять классы в заголовочном файле и реализовать класСы в отдельном файле. Заголовочные файлы (Interface Definition Language - язык описания интерфейсов) и сложные интерфейсы не являются необходимыми.

В C++ переменные экземпляров объектов должны быть явно инициализированы в конструкторе. В C# это делается заранее. Для массива, размер которого [n-1], в С++ можно устанавливать ссылки на это нереальное значение n - то есть выходить за границы массива. В C# это недопустимо. С++ поддерживает множественное наследование классов. Это приводит к определенному усложнению коду. C# не поддерживает множественного наследования. Оно может быть реализовано посредством интерфейсов. В C# все классы имеют единый базовый класс. Если класс не является производным от другого класса, он автоматически наследует свойства и члены из System.Object. Код программ на С/С++ компилируется в формат машинного языка, который будет выполнятся на определенном процессоре и в определенной ОС. Для C# предусмотрена компиляция программ на промежуточный язык IL (Intermediate Language). Код IL может запускаться на любой платформе при использовании среды .NET. С помощью JIT-компилятора он может быть преобразован в машинно-зависимый код для выполнения на платформе .NET

# **6.Анализ характеристик альтернативных программных комплексов**

**Mathcad** содержит сотни операторов и встроенных функций для решения различных технических задач. Программа позволяет выполнять численные и символьные вычисления, производить операции с скалярными величинами, векторами и матрицами, автоматически переводить одни единицы измерения в другие.

Среди возможностей Mathcad можно выделить:

* Решение дифференциальных уравнений, в том числе и численными методами
* Построение двумерных и трёхмерных графиков функций (в разных системах координат, контурные, векторные и т. д.)
* Использование греческого алфавита как в уравнениях, так и в тексте
* Выполнение вычислений в символьном режиме
* Выполнение операций с векторами и матрицами
* Символьное решение систем уравнений
* Аппроксимация кривых
* Выполнение подпрограмм
* Поиск корней многочленов и функций
* Проведение статистических расчётов и работа с распределением вероятностей
* Поиск собственных чисел и векторов
* Вычисления с единицами измерения
* Интеграция с САПР системами, использование результатов вычислений в качестве управляющих параметров

С помощью Mathcad инженеры могут документировать все вычисления в процессе их проведения.

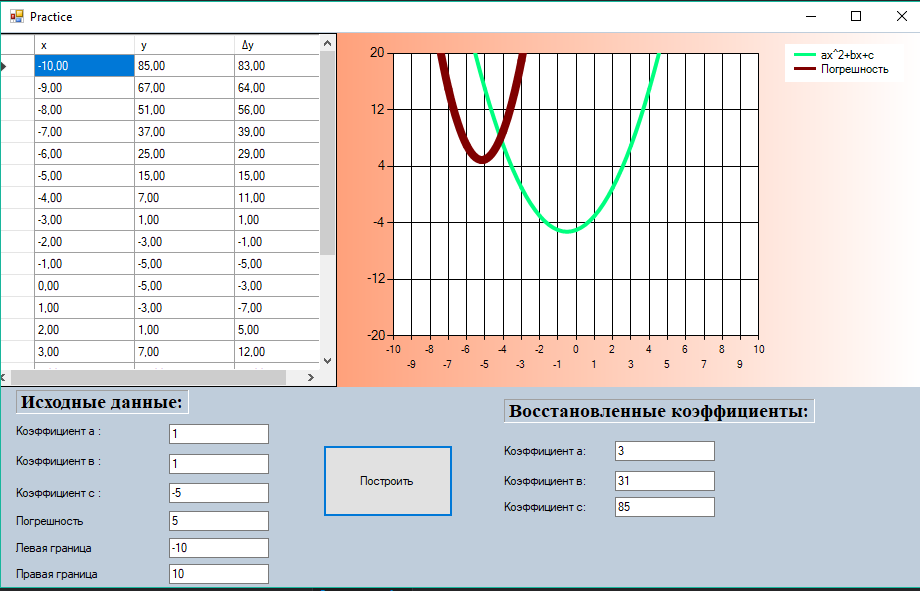
**Scilab** содержит сотни математических функций, и есть возможность добавления новых, написанных на различных языках (C, C++, Fortran и т. д.). Также имеются разнообразные структуры данных (списки, полиномы, рациональные функции, линейные системы), интерпретатор и язык высокого уровня.

Scilab был спроектирован как открытая система, и пользователи могут добавлять в него свои типы данных и операции путём перезагрузки.

В системе доступно множество инструментов:

* 2D и 3D графики, анимация
* Линейная алгебра, [разреженные матрицы](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D1%80%D0%B5%D0%B6%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D1%80%D0%B8%D1%86%D0%B0) (sparse matrices)
* Полиномиальные и рациональные функции
* Интерполяция, аппроксимация
* Симуляция: решение ОДУ и ДУ
* Scicos: гибрид системы моделирования динамических систем и симуляции
* Дифференциальные и не дифференциальные оптимизации
* Обработка сигналов
* Параллельная работа
* Статистика
* Работа с [компьютерной алгеброй](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B0%D0%BB%D0%B3%D0%B5%D0%B1%D1%80%D0%B0)
* Интерфейс к Fortran, [Tcl](https://ru.wikipedia.org/wiki/Tcl" \o "Tcl)/[Tk](https://ru.wikipedia.org/wiki/Tk), [C](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)), [C++](https://ru.wikipedia.org/wiki/C%2B%2B), [Java](https://ru.wikipedia.org/wiki/Java), [LabVIEW](https://ru.wikipedia.org/wiki/LabVIEW)

# **7. Интерфейс программы**



В данном интерфейсе реализовано:

* Вывод параметров
* Графики экспериментальные и исходных данных, аналитически вычисленный
* Аргументы, погрешность функции и сама функция

# **8. План тестирования программного комплекса**

При проведении тестирования, необходимо учесть следующие ситуации:

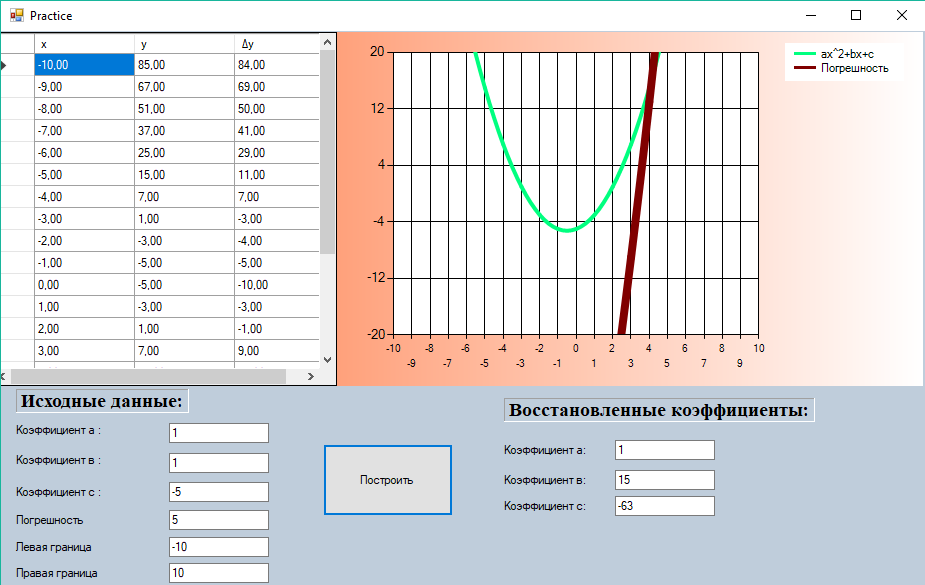
* Изменение значений функции
* Изменение погрешности вычислений

В соответствии с данными ситуациями было спланированно проведение тестирования программного комплекса.

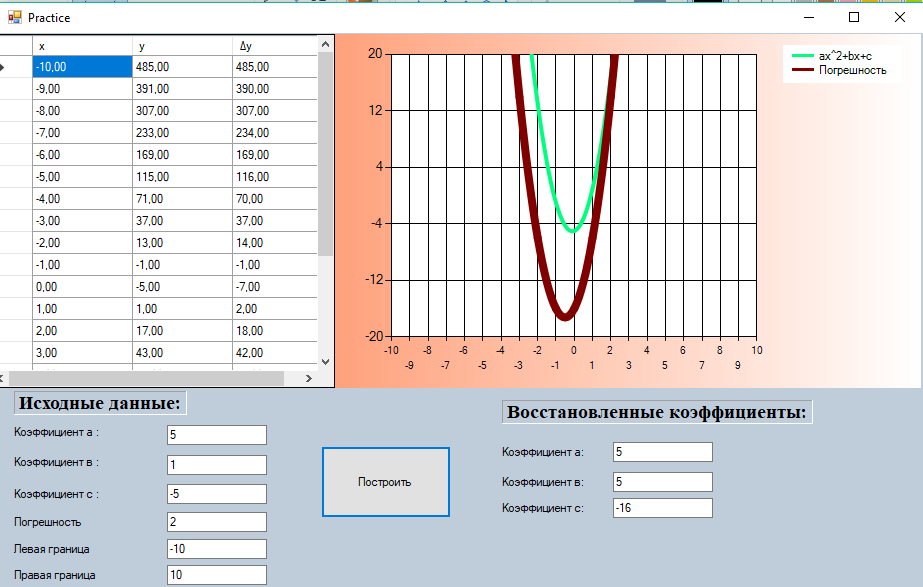
# **9. Тестирование программного комплекса**

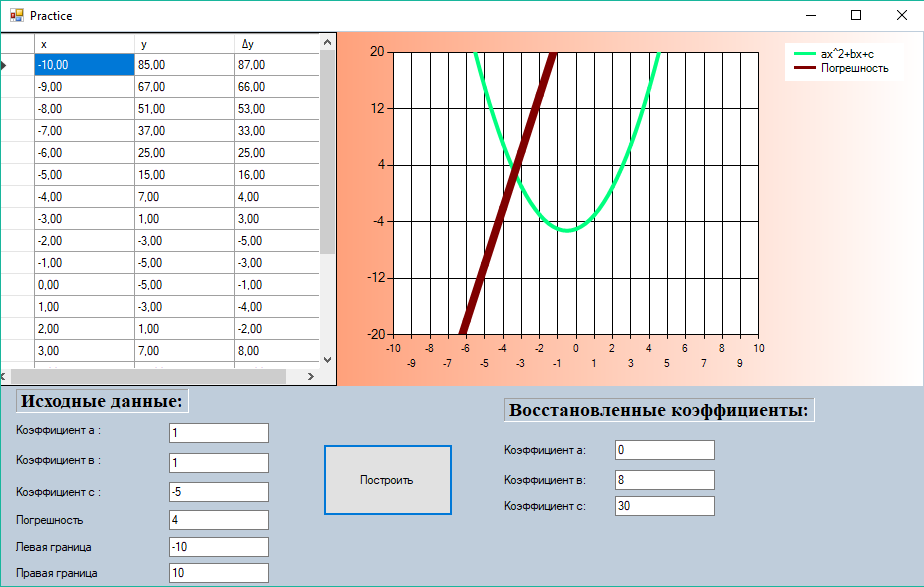
При проведении тестирования получены следующие результаты:

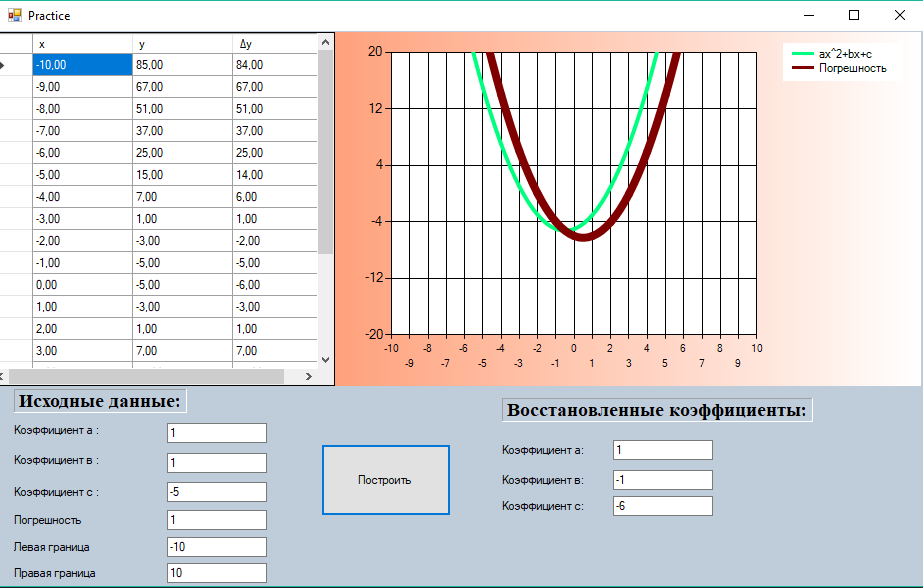
1. Результат тестирования при погрешности равной 5. Как мы видим, экспериментально восстановленная функция кардинально отличается от расчётой.



1. Результат тестирования при погрешности равной 2. Данная погрешность является допустимой.



1. Результат тестирования при погрешности равной 4. Функция восстановилась в линейную зависимость. Из этого можно сделать вывод о негодности экспериментальных данных.
2. Результат тестирования при погрешности равной 1. Восстановленная функция идентична расчётной, с небольшим отклонением.



По проведенному тестированию можно определить, что программный комплекс работает корректно и учитывает возможность изменения всех входных параметров.

# **9. Анализ выполнения учебной практики**

Наиболее сложными моментами стало:

* Поиск информации по восстановлению функции
* Преобразование этих данных в код
* Поиск ошибок
* Проведение корректного тестирования, с учетом всех возможных ситуаций

# **10. Вывод по учебной практике**

Несмотря на все трудности, возникающие в процессе разработки, был получен готовый программный продукт. В чем поспособствовало трудолюбие и знания, перенятые у преподавателей.

# **11. Заключение**

В результате прохождения учебной практики были приобретены новые знания и умения. Развились навыки по поиску необходимой информации и ее обработке. Повысился навык программирования, тем самым опыт, позволяющий повысить свою специализацию. Самым ценным в данной практике я считаю опыт работы над интересным и полезным проектом.

# **ПРИЛОЖЕНИЕ А**

# Текст программы

using System;

using System.Windows.Forms;

namespace Praktika

{

public partial class MainWindow : Form

{

private static double Coef\_A;

private static double Coef\_B;

private static double Coef\_C;

private static int Fault;

private static int LeftBorder;

private static int RigthBorder;

public MainWindow()

{

InitializeComponent();

}

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

{

try

{

Coef\_A = Double.Parse(Coef\_A\_TextBox.Text);

Coef\_B = Double.Parse(Coef\_B\_TextBox.Text);

Coef\_C = Double.Parse(Coef\_C\_TextBox.Text);

Fault = int.Parse(Fault\_TextBox.Text);

LeftBorder = int.Parse(LeftBorder\_TextBox.Text.ToString());

RigthBorder = int.Parse(RigthBorder\_TextBox.Text.ToString());

if (Fault < 0) throw new Exception("Погрешность не может быть отрицательной!");

}

catch (Exception ex)

{

MessageBox.Show("Ошибка! Неверно введены данные! " + ex.Message);

return;

}

GraphicFunction.Series[0].Points.Clear();

GraphicFunction.Series[1].Points.Clear();

GraphicFunction.ChartAreas[0].AxisX.Minimum = LeftBorder;

GraphicFunction.ChartAreas[0].AxisX.Maximum = RigthBorder;

int num\_row = 0;

MainTable.Rows.Clear();

MainTable.RowCount = 0;

GausMethod Solution = new GausMethod(3, 3);

Random rnd = new Random();

bool fl = false;

for (int i = LeftBorder, k\_first = 0, k\_second = 0; i <= RigthBorder; i++)

{

int y = Convert.ToInt32(Algorithms.Calculation\_Parabola(Coef\_A, Coef\_B, Coef\_C, i));

GraphicFunction.Series[0].Points.AddXY(i, y);

double max = y + Fault;

double min = y - Fault;

int fault\_y = Convert.ToInt32(rnd.NextDouble() \* (max - min) + min);

GraphicFunction.Series[1].Points.AddXY(i, fault\_y);

MainTable.RowCount++;

MainTable[0, num\_row].Value = i;

MainTable[1, num\_row].Value = y;

MainTable[2, num\_row].Value = fault\_y;

if (k\_first == 2 || fl)

{

for (int j = 0; j < 3; j++)

{

Solution.RightPart[k\_first] = double.Parse(MainTable[2, num\_row].Value.ToString()) - j;

Solution.Matrix[k\_first][0] = j \* j;

Solution.Matrix[k\_first][1] = j;

Solution.Matrix[k\_first][2] = 1;

}

Solution.SolveMatrix();

MainTable[3, k\_second].Value = Convert.ToInt32(Solution.Answer[0]);

k\_first = 0;

k\_second++;

fl = true;

}

else

{

Solution.RightPart[k\_first] = fault\_y;

Solution.Matrix[k\_first][0] = i \* i;

Solution.Matrix[k\_first][1] = i;

Solution.Matrix[k\_first][2] = 1;

k\_first++;

}

num\_row++;

}

MainTable.Columns[0].DefaultCellStyle.Format = "n2";

MainTable.Columns[1].DefaultCellStyle.Format = "n2";

MainTable.Columns[2].DefaultCellStyle.Format = "n2";

MainTable.Columns[3].DefaultCellStyle.Format = "n2";

BuildToGraphicFunctionDistribution();

}

private void BuildToGraphicFunctionDistribution()

{

double x\_count = 0;

GraphicFault.Series[0].Points.Clear();

GraphicFault.ChartAreas[0].AxisY.Maximum = 1;

double m = MainTable.RowCount;

double high\_function = 1 / m;

double count\_function = 0;

for (int i = 0; i < m - 2; i++)

{

double j = FindCountElements(i);

count\_function += j / m;

GraphicFault.Series[0].Points.AddXY(count\_function, high\_function);

high\_function += 1 / m;

x\_count++;

}

}

private void MainTable\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private int FindCountElements(int count)

{

int num = 0;

int el = Convert.ToInt32(MainTable[3, count].Value.ToString());

for (int j = 0; j < MainTable.RowCount - 2; j++)

{

if (Convert.ToInt32(MainTable[3, j].Value.ToString()) <= el) num++;

}

return num;

}

}

}

using System;

namespace Praktika

{

class Algorithms

{

public static double Calculation\_Parabola(double a, double b, double c, double x)

{

return a \* Math.Pow(x,2) + b \* x + c;

}

class GausMethod

{

public uint RowCount;

public uint ColumCount;

public double[][] Matrix { get; set; }

public double[] RightPart { get; set; }

public double[] Answer { get; set; }

public GausMethod(uint Row, uint Colum)

{

RightPart = new double[Row];

Answer = new double[Row];

Matrix = new double[Row][];

for (int i = 0; i < Row; i++)

Matrix[i] = new double[Colum];

RowCount = Row;

ColumCount = Colum;

//обнулим массив

for (int i = 0; i < Row; i++)

{

Answer[i] = 0;

RightPart[i] = 0;

for (int j = 0; j < Colum; j++)

Matrix[i][j] = 0;

}

}

private void SortRows(int SortIndex)

{

double MaxElement = Matrix[SortIndex][SortIndex];

int MaxElementIndex = SortIndex;

for (int i = SortIndex + 1; i < RowCount; i++)

{

if (Matrix[i][SortIndex] > MaxElement)

{

MaxElement = Matrix[i][SortIndex];

MaxElementIndex = i;

}

}

//теперь найден максимальный элемент ставим его на верхнее место

if (MaxElementIndex > SortIndex)//если это не первый элемент

{

double Temp;

Temp = RightPart[MaxElementIndex];

RightPart[MaxElementIndex] = RightPart[SortIndex];

RightPart[SortIndex] = Temp;

for (int i = 0; i < ColumCount; i++)

{

Temp = Matrix[MaxElementIndex][i];

Matrix[MaxElementIndex][i] = Matrix[SortIndex][i];

Matrix[SortIndex][i] = Temp;

}

}

}

public int SolveMatrix()

{

if (RowCount != ColumCount)

return 1; //нет решения

for (int i = 0; i < RowCount - 1; i++)

{

SortRows(i);

for (int j = i + 1; j < RowCount; j++)

{

if (Matrix[i][i] != 0) //если главный элемент не 0, то производим вычисления

{

double MultElement = Matrix[j][i] / Matrix[i][i];

for (int k = i; k < ColumCount; k++)

Matrix[j][k] -= Matrix[i][k] \* MultElement;

RightPart[j] -= RightPart[i] \* MultElement;

}

//для нулевого главного элемента просто пропускаем данный шаг

}

}

//ищем решение

for (int i = (int)(RowCount - 1); i >= 0; i--)

{

Answer[i] = RightPart[i];

for (int j = (int)(RowCount - 1); j > i; j--)

Answer[i] -= Matrix[i][j] \* Answer[j];

if (Matrix[i][i] == 0)

if (RightPart[i] == 0)

return 2; //множество решений

else

return 1; //нет решения

Answer[i] /= Matrix[i][i];

}

return 0;

}

public override String ToString()

{

String S = "";

for (int i = 0; i < RowCount; i++)

{

S += "\r\n";

for (int j = 0; j < ColumCount; j++)

{

S += Matrix[i][j].ToString("F04") + "\t";

}

S += "\t" + Answer[i].ToString("F08");

S += "\t" + RightPart[i].ToString("F04");

}

return S;

}

}

}

}

namespace Praktika

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new MainWindow());

}

}

}