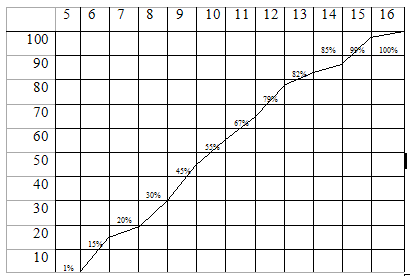
ФГБОУ ВПО

«Уфимский государственный авиационный технический университет»

**Кафедра Информатики**



|  |
| --- |
| Тема КР |
| Решение систем |
| линейных уравнений |

## ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

|  |  |
| --- | --- |
| **к курсовой работе по** | **информатике** |

|  |
| --- |
| 1308.301202.000ПЗ |
| (обозначение документа) |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Группа |  |  | Фамилия, и., о. | Подпись | Дата | Оценка |
| ИВТ-109 |  |
|  |  |
| Студент | | | Вафин Р. Р. |  |  |  |
| Консультант | | | Смирнова Е.А. |  |  |  |
| Принял | | |  |  |  |  |

**Уфа 2015 г.**

**Содержание**

[1.1 ПОЯСНИТЕЛЬНАЯ ЗАПИСКА 1](#_Toc418071776)

[1 Введение 3](#_Toc418071777)

[1.1 Класс задач 3](#_Toc418071778)

[1.2 Примеры задач 4](#_Toc418071779)

[2 Описание метода 7](#_Toc418071780)

[2.1 Метод Гаусса 7](#_Toc418071781)

[2.2 Блок-схема 9](#_Toc418071782)

[2.4 Код программы 10](#_Toc418071783)

[3 Описание программы 12](#_Toc418071784)

[3.1 Общая структура программы 12](#_Toc418071785)

[3.2 Main\_form 13](#_Toc418071786)

[3.3 Menu\_form 13](#_Toc418071787)

[3.4 Form1 15](#_Toc418071788)

[3.5 VisForm 17](#_Toc418071789)

[3.6 Description\_form 20](#_Toc418071790)

[3.7 Flowchart\_form 20](#_Toc418071791)

[3.8 Ref\_form 21](#_Toc418071792)

[3.9 About\_form 22](#_Toc418071793)

[4 Вывод 23](#_Toc418071794)

[5 Используемая литература 25](#_Toc418071795)

[Приложение 26](#_Toc418071796)

# Введение

## Класс задач

Системы линейных алгебраических уравнений (СЛАУ) применяются в самых разных областях науки. Это может быть электротехника, механика, термодинамика, информатика и тд.

В моей практике СЛАУ встречается в физике (Правила Кирхгофа), в теоретических основах электротехники, ну и главным образом в математическом анализе, где СЛАУ одна из основных тем 1-го семестра.

Задачи на правила Кирхгофа – это задача на нахождение соотношения тока и напряжения на участках любой электрической цепи. С этими задачами сталкиваются на курсах общей физики, на теоретических основах электротехники, а так же те, кто самостоятельно разрабатывает электрические устройства. Применение правил Кирхгофа к линейной электрической цепи позволяет получить систему линейных уравнений относительно токов или напряжений, и соответственно, найти значение токов на всех ветвях цепи и все меж узловые напряжения.

Так же довольно обширной областью применения СЛАУ является экономика. К таким типам задач можно отнести: прогноз и оценки экономики предприятий, организаций; прогноз курса валют, ценных бумаг, акций, и тд.

Прогноз погоды – так же является одной из целью применения СЛАУ. Обширные объемы данных (температура, давление, влажность и тд) требуют большой вычислительной мощности, и для решения этих систем применяются наиболее эффективные алгоритмы решения СЛАУ, а так же большые вычислительные мощности.

Весь этот спектр задач предопределяет методы, необходимые для их решения. Так к примеру простые задачи на курсах физики или математического анализа требуют легкого, простого для понимания метода решения, а задачи связанные с прогнозом погоды требуют большей эффективности, и в следствии сложных, но быстрых алгоритмов решения СЛАУ.

В данной работе я буду рассматривать метод Гаусса, как самый простой для использования, и быстрый для СЛАУ небольшого порядка. Но при порядке систем более 30-40 решения может занимать довольно продолжительное время, и вследствие этого, моя программа предназначения для решения СЛАУ только до 10-го порядка.

Ниже предоставлены задачи, которые можно решить, используя мою программу.

## Примеры задач

**Задача на правила Кирхгофа. (**Методические указания к выполнению курсовых работ по дисциплине «Информатика»**)**

Проанализировать работу электрической цепи из линейных элементов по приведенной многоконтурной схеме (Рисунок 1.2.1): найти неизвестные токи i1, i2, i3 по заданным значениям сопротивлений R1, R2, R3 и электродвижущей силе источника тока E.

Исходными данными для решения этой задачи являются: значения сопротивлений и электродвижущая сила источника тока. Результатом являются значения токов.

Все величины представлены в системе измерения СИ.

Все величины – вещественного типа.



Рисунок 1.1 Многоконтурная схема электрической цепи к задаче.

Решением данной задачи будет система:

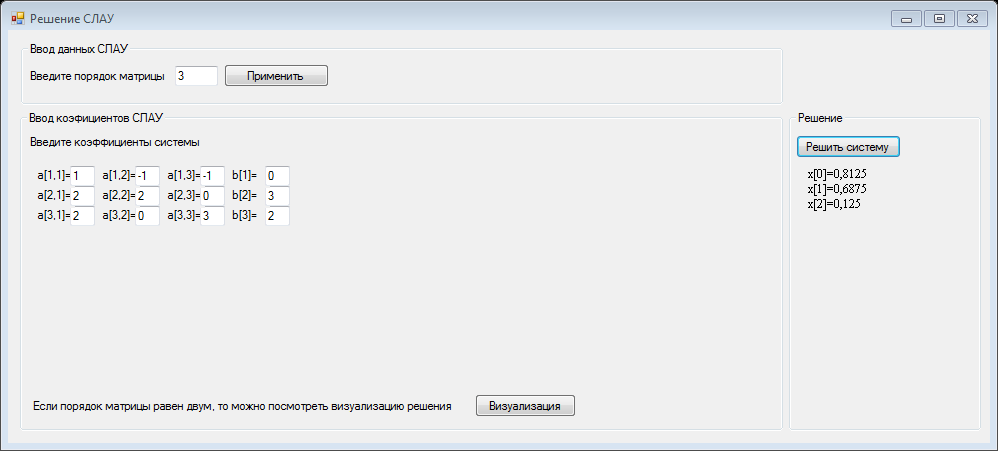


Преобразуя эту систему в матричном виде получим:



Используя программу можно решить эту задачу. Значения сопротивлений и напряжений возьмем равными: R1=2 R2=2 R3=3 E=2.

Решением будет: i1=0.8125 i2=0,6875 i3=0,125.



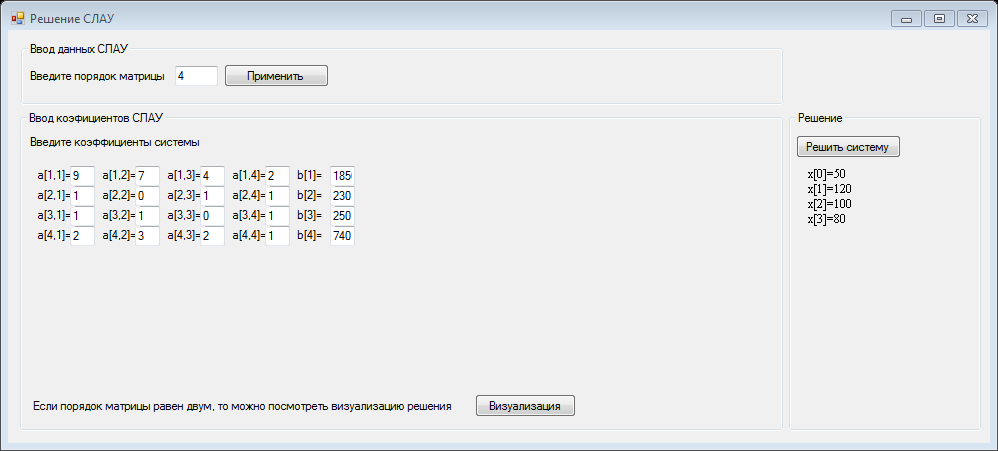
**Задача, связанная с экономикой. (ниже)**

На предприятие с работниками четырех категорий привезли заработную плату в купюрах следующего достоинства: по 100 рублей – 1850 купюр, по 10 рублей – 250 купюр, 1 рублю – 740 купюр. Заработная плата работника 1-й категории составляет 962 руб., 2-й категории –713 руб., 3-й категории – 452 руб., 4-й категории – 261 руб. Определить, сколько сотрудников каждой категории работает на предприятии, если каждому сотруднику выдали заработную плату минимальным числом купюр.

Пусть *x*, *y*, *z*, *w* – количество работников категорий соответственно с первой по четвертую. Тогда решением данной задачи будет система уравнений:



И ее решение: x=50, y=120, z=100, w=80



# Описание метода

Система линейных уравнений обычно записывается в виде:

 (2.1)

В матричном виде система линейных уравнений записывается так:



где

## Метод Гаусса

Метод Гаусса (метод исключения) для решения систем линейных уравнений относится к точным методам. Идея метода Гаусса состоит в том, что система (2.1) путем последовательного исключения неизвестных приводится к системе с треугольной матрицей, из которой и определяются значения неизвестных.

Процесс исключения неизвестных состоит в следующем:

Пусть a11≠0. Разделим первое уравнение на a11. Затем вычтем из каждого i-го (i≥2) уравнения, полученного после деления, первое, умноженное на ai1. В результате, после преобразований x1 окажется исключенным из всех уравнений кроме первого. По той же схеме исключается x2 (разделив второе уравнение на a22≠0), затем x3 и т.д.

В результате получается треугольная матрица с единичной главной диагональю.

 (2.2)

Особенность этой системы – в строках с номером i все коэффициенты aij при j<i равны нулю. Эту систему уравнений треугольного вида решить уже просто. Из последнего уравнения определяется xn, далее, подставляя его в предпоследнее уравнение, получаем xn-1 и т.д.

Общая формула определения неизвестных имеет вид

 (2.3)

Приведение системы (2.1) к треугольному виду (2.2) называется *прямым ходом метода Гаусса.* Процесс исключения k-го неизвестного называется *k-м шагом прямого хода*. Элементы  называются *ведущими*.

Общие формулы пересчета коэффициентов системы на k-м шаге имеют вид:







Определение неизвестных по формулам (2.3) называется обратным ходом метода Гаусса.

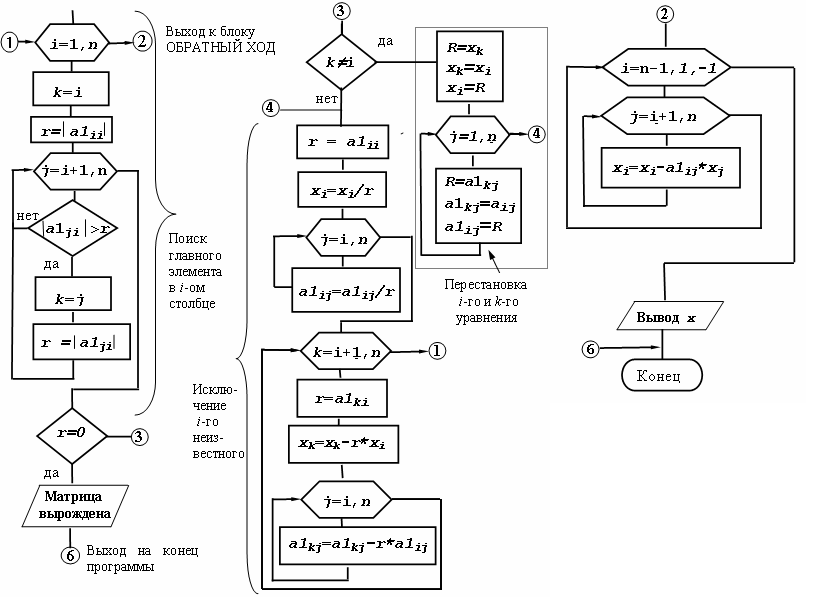
В методе Гаусса происходит деление строк на соответствующие ведущие элементы, поэтому, если на каком-то *k*-м шаге на главной диагонали окажется нулевой элемент  то среди элементов  (i=k+1,…, n) следует найти ненулевой и перестановкой строк переместить его на главную диагональ, а затем продолжить вычисления.

Для этого следует воспользоваться, например, методом Гаусса выбора главного элемента в столбце, суть которого состоит в определении максимального элемента в столбце текущей строки и перестановке строки с максимальным элементом в столбце с текущей строкой, если таковой найден.

Если такого ненулевого элемента не найдется, то определитель системы равен нулю и система либо не имеет решений, либо решений бесконечно много.

На рис. 2.1 представлена блок-схема прямого хода – исключение *i*-го неизвестного по методу Гаусса. На рис. 2.2 представлена блок-схема обратного хода – определение неизвестных по методу Гаусса

## Блок-схема



## Код программы

public double[] solve(double[,] A, double[] X, int N)

{

double r = 0, R = 0;

int i, j, k;

//

for (i = 0; i < N; i++)

{

k = i;

r = A[i, i];

for (j = i + 1; j < N; j++)

{

if (A[j, i] > r)

{

k = j;

r = A[j, i];

}

}

if (r == 0)

{

break;

}

if (k != i)

{

R = X[k];

X[k] = X[i];

X[i] = R;

for (j = 0; j < N; j++)

{

R = A[k, j];

A[k, j] = A[i, j];

A[i, j] = R;

}

}

r = A[i, i];

X[i] = X[i] / r;

for (j = i; j < N; j++)

A[i, j] = A[i, j] / r;

for (k = i + 1; k < N; k++)

{

r = A[k, i];

X[k] = X[k] - r \* X[i];

for (j = i; j < N; j++)

A[k, j] = A[k, j] - r \* A[i, j];

}

}

if (r != 0)

{

for (i = N - 2; i >= 0; i--)

for (j = i + 1; j < N; j++)

X[i] = X[i] - A[i, j] \* X[j];

return X;

}

else

{

throw new System.Exception();

}

}

# Описание программы

## Общая структура программы

Программа для решения СЛАУ написана на языке программирования C# в интегрированной среде разработки Microsoft Visual Studio 2013.

Программа состоит из 7 форм:

1. Main\_form – Титульный лист. Появляется при запуске программы.
2. Menu\_form – Основное меню. Из него можно вызвать все остальные окна.
3. Form1 – Окно решения СЛАУ
4. VisForm – Окно визуализации.
5. About\_form – Окно «О программе».
6. Ref\_form – Справка.
7. Description\_form – Описание метода Гаусса.
8. Flowchart\_form – Блок-схема метода Гаусса.

Визуально структуру программы можно предоставить в виде графа, который показывает взаимодействие их между собой.

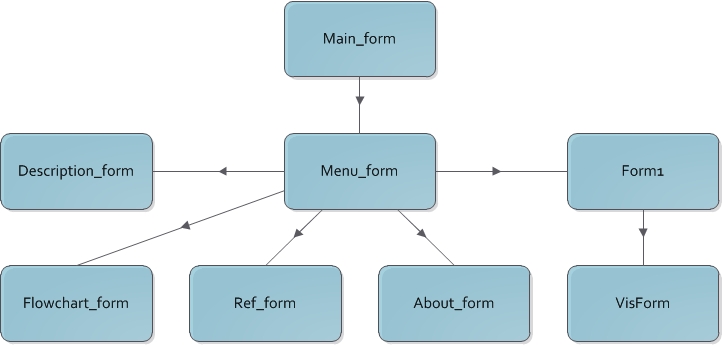


Рисунок 3.1 Структура программы

## Main\_form

рлрллр

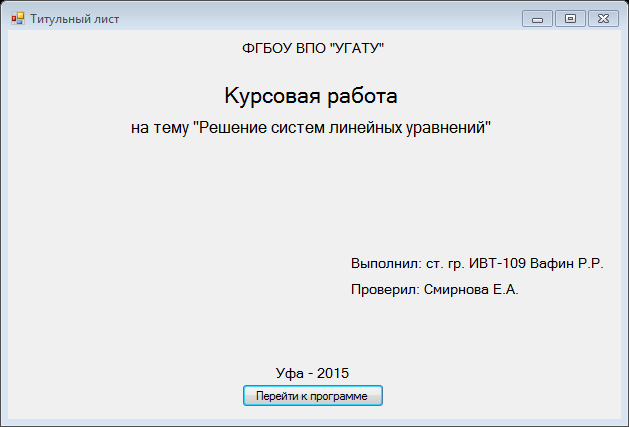


Рисунок 3.2 Титульный лист.

Это титульный лист. Логика работы очень проста, при нажатии кнопки «Перейти к программе» срабатывает обработчик события (private void button1\_Click(object sender, EventArgs e)), который создает и делает видимой форму Menu\_form, и делает невидимой Main\_form.

## Menu\_form

Основная форма с меню.

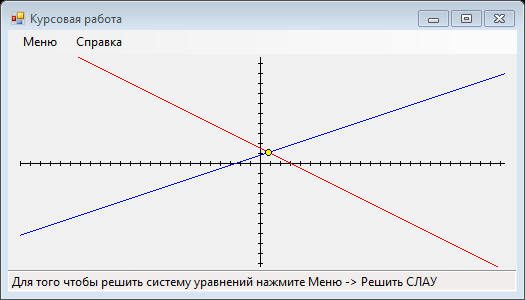


Рисунок 3.3 Основное меню.

**Пункты меню:**

1. Меню
   1. Титульный лист (private void TL\_menuItem\_Click(object sender, EventArgs e)) – открывает Титульный лист.
   2. Описание метода
      1. Описание (private void Desc\_menuItem\_Click(object sender, EventArgs e)) – открывает описание метода.
      2. Блок-схема (private void FC\_menuItem\_Click(object sender, EventArgs e)) – открывает блок-схему метода .
   3. Решить СЛАУ (private void Solve\_menuItem\_Click(object sender, EventArgs e)) – открывает основную форму для решения СЛАУ.
   4. Выход (private void Menu\_form\_FormClosed(object sender, FormClosedEventArgs e)) – закрывает программу.
2. Справка
   1. Справка (private void Ref\_menuItem\_Click(object sender, EventArgs e)) – открывает справку программы.
   2. О программе (private void About\_nemuItem\_Click(object sender, EventArgs e)) – открывает окно с названием программы, и ее автором.

Каждый из пунктов меню имеет свой обработчик события нажатия, который указаны выше. В каждом из них создается новая форма, и делается видимой.

## Form1

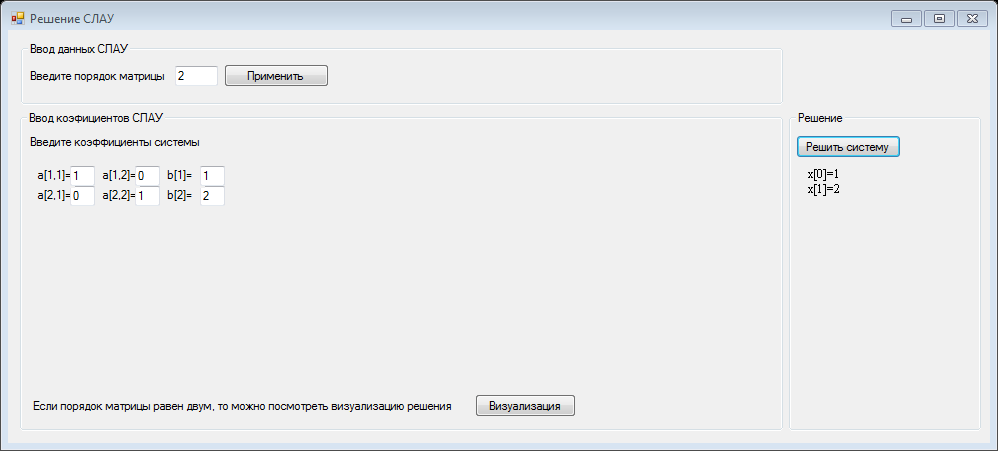


Рисунок 3.4 Форма решения СЛАУ.

Основная форма для решения СЛАУ.

Для того, чтобы решить СЛАУ, необходимо:

1. Ввести порядок матрицы.
2. Нажать кнопку «Применить» (private void Init\_Input\_Field(object sender, EventArgs e) ). Если порядок больше 10, то выдаст сообщение о ошибке.
3. Ввести коэффициенты системы в поле ввода.
4. Нажать кнопку «Решить систему» (private void Solve\_SLAU(object sender, EventArgs e) ).
5. Если порядок введённой системы равен двум, то чтобы посмотреть визуализацию решения системы нужно нажать кнопку «Визуализация» (private void btn\_vis(object sender, EventArgs e)), иначе выдаст сообщение о ошибке.

Примеры возможностей программы:

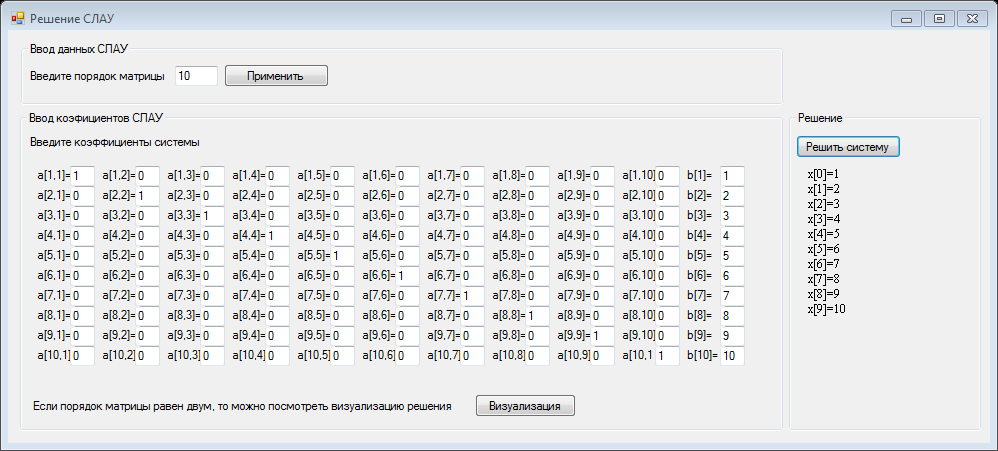


Рисунок 3.5 Решение системы 10-го порядка.

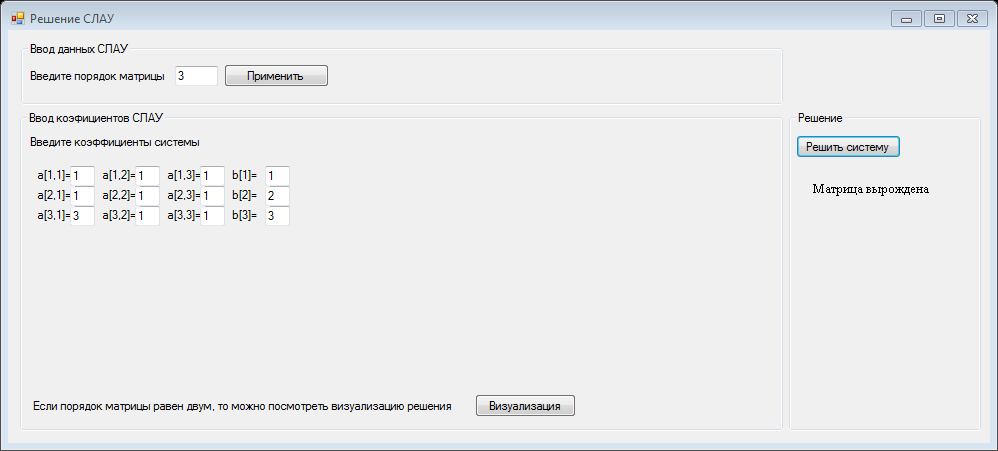


Рисунок 3.6 Сообщение о том, что система не имеет решения.

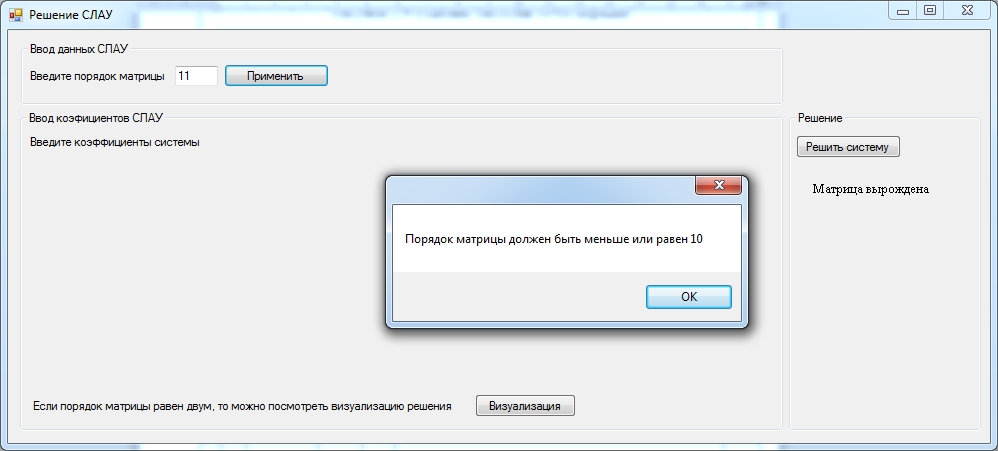


Рисунок 3.7 Сообщение о недопустимом порядке системы.

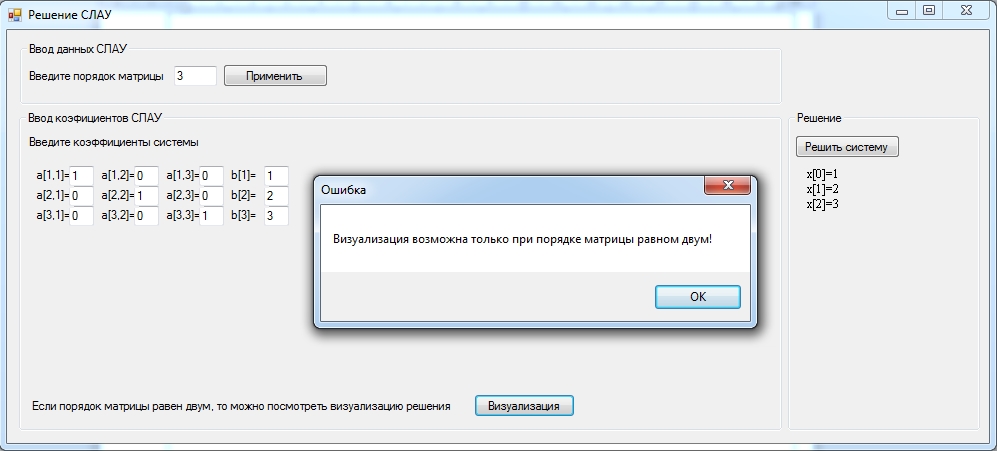


Рисунок 3.8 Сообщение о невозможности визуализации при неверном порядке.

## VisForm

Форму визуализации можно открыть только для систем 2-го порядка. Она представляет собой декартову систему координат, на которой изображены 2 линии – графики линейных функций. Их взаимное расположение указывается в левом верхнем углу формы. Визуализация осуществляется с помощью данного метода: (public void InitMatrix(double[,] A, double[] B)).

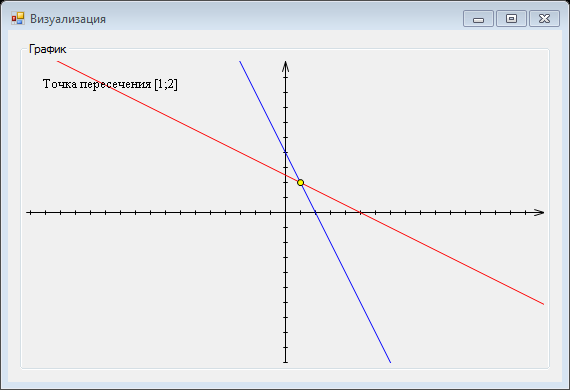


Рисунок 3.9 Визуализация решения системы.

На рисунке 3.8 показана визуализация решения системы 2-го порядка. Это 2 прямые, координаты точки пересечения который и есть решение данной системы. Если же система не имеет решения, то возможны 2 варианта:

1. Графики параллельны, и тогда решения нет вообще. Рисунок 3.10
2. Графики накладываются друг на друга, и тогда система имеет бесконечное количество решений. Рисунок 3.9

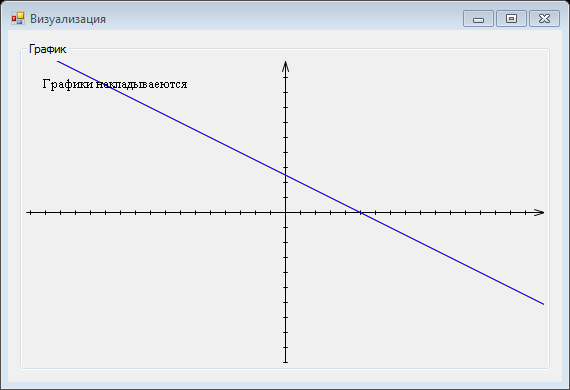


Рисунок 3.10 Графики накладываются .

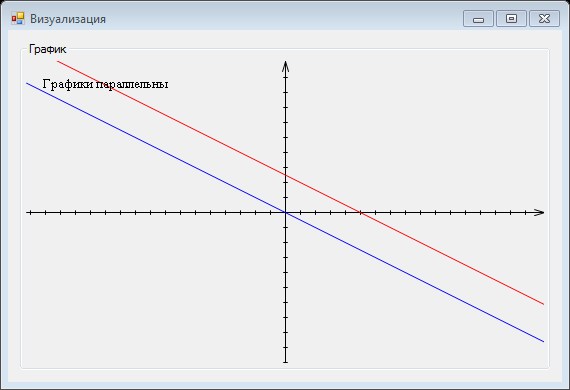


Рисунок 3.11 Графики параллельны.

## Description\_form

Форма описания метода Гаусса. В ней описан метод гаусса.

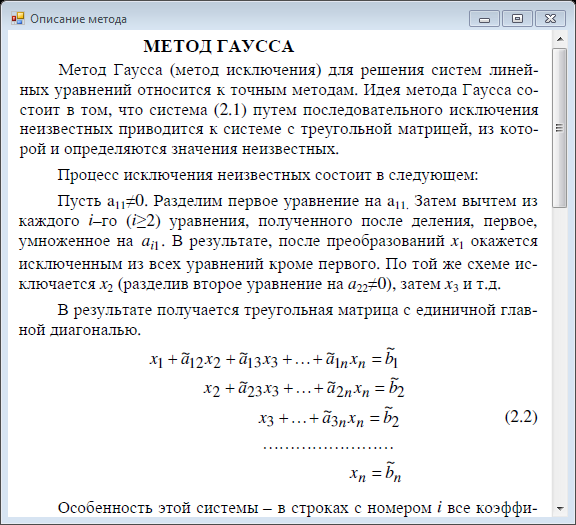


Рисунок 3.12 Форма описание метода.

## Flowchart\_form

Форма блок-схемы метода гаусса.

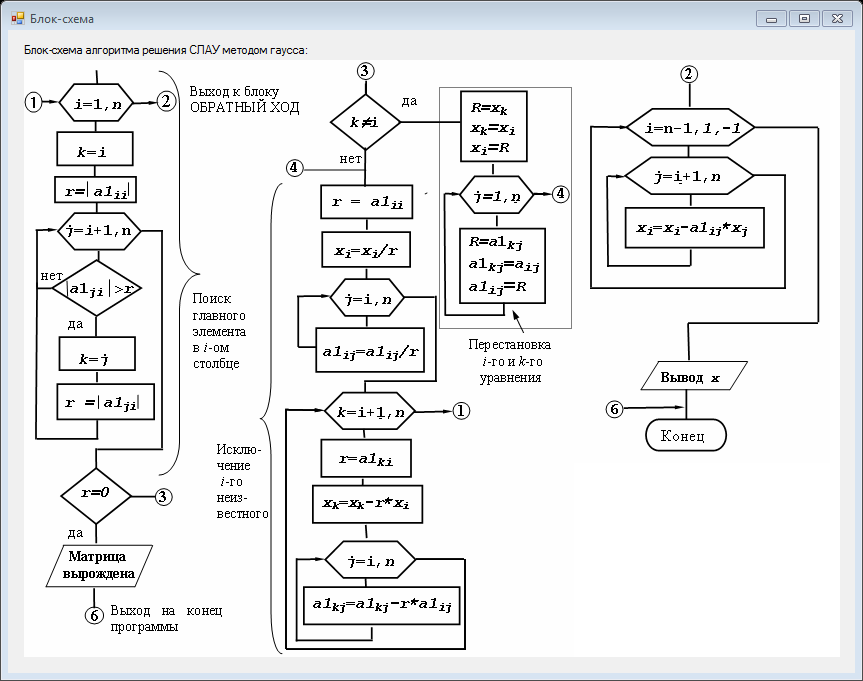


Рисунок 3.13 Форма блок-схемы метода Гаусса.

## Ref\_form

Форма справки. Там описано как работать с программой. Описание всех элементов управления.

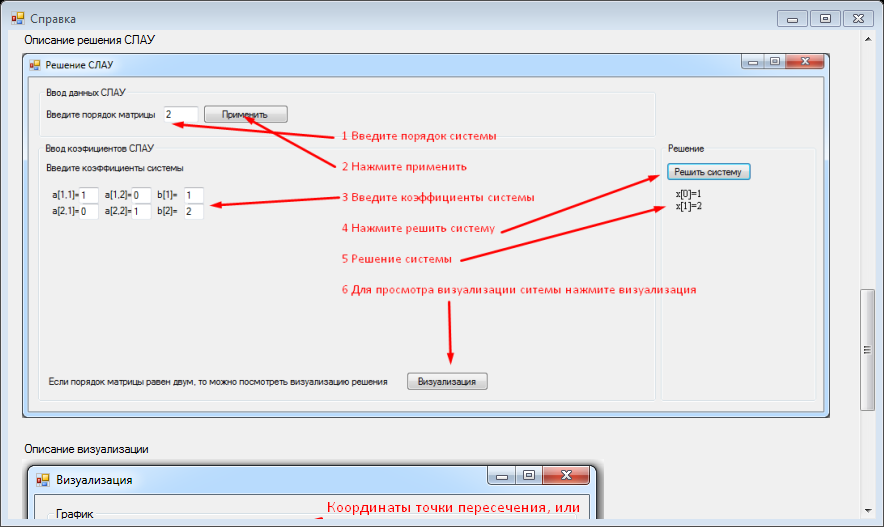


Рисунок 3.14 Справка

## About\_form

Форма «О программе». Указаны название программы, логотип программы, а так же автор программы.

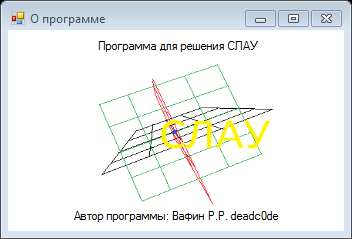


Рисунок 3.15 О программе.

# Вывод

В ходе выполнения курсовой работы были изучены базовые возможности интегрированной среды разработки (IDE) Microsoft Visual Studio 2013 для разработки на языке высокого уровня C#, изучены основы языке С# для разработки многооконных приложений, создания ветвящихся, циклических, и рекурсивных алгоритмов, изучены основы ООП (Объектно-Ориентированного Программирования).

Самым большим препятствие в работе над курсовой работой было недостаточное знание языка программирования C#. По этому в начале работы часть логики работы программы были написаны с использованием, так называемых, «костылей» - участков кода, которые хоть и решают поставленную задачу, но делают это нестандартным образом. К примеру, можно отнести реализацию закрытия программы из неосновной формы. Так как в то время мне не было известно как правильно закрывать приложение, я из другой формы, удалял основную, и тогда программа закрывалась. В ходе работы большая часть «костылей» была убрана. Язык C# изучался по «Методические указания к выполнению курсовых работ по дисциплине «Информатика»», «Герберт Шилдт - C# 4.0. Полное руководство» а так же по «Каталог API (Microsoft) и справочных материалов»

Данная программа может применяться студентами младших курсов для упрощения работы с математическим анализом, так как СЛАУ – это одна из тем первого семестра, с физикой и другими предметными областями, в которых, для решения задач, необходимо решать системы уравнений. Возможностей программы хватит для решения практически любой задачи на СЛАУ из первого курса математического анализа, так как там системы больше десятого порядка не рассматриваются.

Подводя итоги можно сказать, что в ходе работы над курсовой работой, я получил большой опыт в создании сложных многооконных приложений на языке C#, и создал программу, которая была бы очень полезной для студентов младших курсов.

# Используемая литература

1. Методические указания к выполнению курсовых работ по дисциплине «Информатика» А.Т. Бикмеев, М.П. Карчевская, Е. А. Кузьмина, О.Л. Рамбургер
2. Герберт Шилдт - C# 4.0. Полное руководство – 2011г

1. <https://msdn.microsoft.com/ru-ru/library/> Каталог API (Microsoft) и справочных материалов
2. <http://studopedia.ru/3_93698_primenenie-sistem-lineynih-uravneniy-v-ekonomike.html>

# Приложение

Form1.cs

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;

namespace Kursovaya

{

public partial class Form1 : Form

{

public int N = 2;

public System.Windows.Forms.Control[,] In\_matrix, In\_matrix\_L;

private Graphics g;

private VisForm vs;

public Form1()

{

InitializeComponent();

Init\_Input\_Field(this,null);

}

public double[] solve(double[,] A, double[] X, int N)

{

double r = 0, R = 0;

int i, j, k;

//

for (i = 0; i < N; i++)

{

k = i;

r = A[i, i];

for (j = i + 1; j < N; j++)

{

if (A[j, i] > r)

{

k = j;

r = A[j, i];

}

}

if (r == 0)

{

break;

}

if (k != i)

{

R = X[k];

X[k] = X[i];

X[i] = R;

for (j = 0; j < N; j++)

{

R = A[k, j];

A[k, j] = A[i, j];

A[i, j] = R;

}

}

r = A[i, i];

X[i] = X[i] / r;

for (j = i; j < N; j++)

A[i, j] = A[i, j] / r;

for (k = i + 1; k < N; k++)

{

r = A[k, i];

X[k] = X[k] - r \* X[i];

for (j = i; j < N; j++)

A[k, j] = A[k, j] - r \* A[i, j];

}

}

if (r != 0)

{

for (i = N - 2; i >= 0; i--)

for (j = i + 1; j < N; j++)

X[i] = X[i] - A[i, j] \* X[j];

return X;

}

else

{

throw new System.Exception();

}

}

private void print\_false()

{

Out\_pic.Image = new Bitmap(Out\_pic.Width, Out\_pic.Height);

Brush b = new SolidBrush(Color.Black);

Font f = new Font("Times New Roman", 10);

g = Graphics.FromImage(Out\_pic.Image);

g.TextRenderingHint = System.Drawing.Text.TextRenderingHint.AntiAliasGridFit;

g.DrawString("Матрица вырождена", f, b, new PointF(15,15));

}

private void print\_X(double[] X)

{

Out\_pic.Image = new Bitmap(Out\_pic.Width, Out\_pic.Height);

g = Graphics.FromImage(Out\_pic.Image);

g.TextRenderingHint = System.Drawing.Text.TextRenderingHint.AntiAliasGridFit;

Brush b = new SolidBrush(Color.Black);

Font f = new Font("Times New Roman", 10);

int H = Out\_pic.Height, W = Out\_pic.Width;

int i;

for (i = 0; i < N; i++)

{

g.DrawString("x[" + Convert.ToString(i) + "]=" + Convert.ToString(X[i]), f, b, new PointF(10, i \* 15));

}

}

private void Init\_Input\_Field(object sender, EventArgs e)

{

int i, j;

if (In\_matrix!=null)

for (i = 0; i < N; i++)

for (j = 0; j < N + 1; j++)

{

In\_matrix[i, j].Dispose();

In\_matrix\_L[i, j].Dispose();

}

N=Convert.ToInt16(Rang.Text);

if (N > 10)

{

N = 10;

MessageBox.Show("Порядок матрицы должен быть меньше или равен 10");

}

In\_matrix = new System.Windows.Forms.Control[N, N + 1];

In\_matrix\_L = new System.Windows.Forms.Control[N, N + 1];

int offset = 40;

for (i = 0; i < N; i++)

for (j = 0; j < N + 1; j++)

{

In\_matrix[i, j] = new System.Windows.Forms.TextBox();

In\_area.Controls.Add(In\_matrix[i, j]);

In\_matrix[i, j].Location = new Point(j \* 65 + 50, i \* 20 + 15 + offset);

In\_matrix[i, j].Size = new Size(25, 20);

In\_matrix[i, j].TabIndex = 4;

if (i == j) In\_matrix[i, j].Text = "1";

else if (j == N) In\_matrix[i, j].Text = Convert.ToString(i+1);

else In\_matrix[i, j].Text = "0";

In\_matrix\_L[i, j] = new System.Windows.Forms.Label();

In\_area.Controls.Add(In\_matrix\_L[i, j]);

In\_matrix\_L[i, j].Location = new Point(j \* 65 + 15, i \* 20 + 17 + offset);

In\_matrix\_L[i, j].Size = new Size(40, 20);

if (j != N) In\_matrix\_L[i, j].Text = "a[" + Convert.ToString(i + 1) + "," + Convert.ToString(j + 1) + "]=";

else In\_matrix\_L[i, j].Text = "b[" + Convert.ToString(i + 1) + "]=";

}

}

private void Solve\_SLAU(object sender, EventArgs e)

{

double[,] A = new double[N, N];

double[] B = new double[N];

double[] X;

int i, j;

try

{

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

A[j, i] = Convert.ToDouble(In\_matrix[j, i].Text);

for (i = 0; i < N; i++)

B[i] = Convert.ToDouble(In\_matrix[i, N].Text);

X = solve(A, B, N);

print\_X(X);

}

catch

{

print\_false();

}

}

private void btn\_vis(object sender, EventArgs e)

{

if (N == 2)

{

int i, j;

double[,] A = new double[N, N];

double[] B = new double[N];

for (i = 0; i < N; i++)

for (j = 0; j < N; j++)

A[j, i] = Convert.ToDouble(In\_matrix[j, i].Text);

for (i = 0; i < N; i++)

B[i] = Convert.ToDouble(In\_matrix[i, N].Text);

vs = new VisForm();

vs.InitMatrix(A, B);

vs.Show();

}

else

MessageBox.Show("Визуализация возможна только при порядке матрицы равном двум!", "Ошибка");

}

}

}

VisForm.cs

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;

namespace Kursovaya

{

public partial class VisForm : Form

{

double[,] AB,X;

double A1, A2, B1, B2, C1, C2;

double k, b;

public VisForm()

{

InitializeComponent();

}

public void InitMatrix(double[,] A, double[] B)

{

int i,j;

Out\_vis.Image = new Bitmap(Out\_vis.Width, Out\_vis.Height);

Graphics g = Graphics.FromImage(Out\_vis.Image);

g.TextRenderingHint = System.Drawing.Text.TextRenderingHint.AntiAliasGridFit;

g.SmoothingMode = System.Drawing.Drawing2D.SmoothingMode.HighQuality;

Pen p = Pens.Black;

Brush br = new SolidBrush(Color.Black);

Font f = new Font("Times New Roman", 10);

float H = Out\_vis.Height, W = Out\_vis.Width;

double Scale = 15;

A1 = A[0, 0]; A2 = A[1, 0];

B1 = A[0, 1]; B2 = A[1, 1];

C1 = B[0]; C2 = B[1];

//grid

int arrow\_a = 3, arrow\_b = 10;

g.DrawLine(p, 0, H / 2, W, H / 2); //x line

g.DrawLine(p, W / 2, 0, W / 2, H); //y line

g.DrawLine(p, W - arrow\_b, H / 2 + arrow\_a, W, H / 2); //x arrow

g.DrawLine(p, W - arrow\_b, H / 2 - arrow\_a, W, H / 2);

g.DrawLine(p, W / 2 - arrow\_a, arrow\_b, W / 2, 0); //y arrow

g.DrawLine(p, W / 2 + arrow\_a, arrow\_b, W / 2, 0);

for (i = 0; i <= W - arrow\_b; i++) //x

{

if ((W / 2 - i) % Scale == 0)

g.DrawLine(p,i,H/2+2,i,H/2-2);

}

for (i = arrow\_b; i <= H; i++) //y

{

if ((H / 2 - i) % Scale == 0)

g.DrawLine(p, W / 2 + 2, i, W / 2 - 2, i);

}

//line1

if (B1 != 0)

{

k = -A1 / B1;

b = C1 / B1;

g.DrawLine(Pens.Red, 0, Convert.ToSingle(-(-W / 2 \* k + b \* Scale) + H / 2),

Convert.ToSingle(W), Convert.ToSingle(-(W / 2 \* k + b \* Scale) + H / 2));

}

else

{

g.DrawLine(Pens.Red, Convert.ToSingle(C1 / A1 \* Scale + W / 2), 0, Convert.ToSingle(C1 / A1 \* Scale + W / 2), H);

}

//line2

if (B2 != 0)

{

k = -A2 / B2;

b = C2 / B2;

g.DrawLine(Pens.Blue, 0, Convert.ToSingle(-(-W / 2 \* k + b \* Scale) + H / 2),

Convert.ToSingle(W), Convert.ToSingle(-(W / 2 \* k + b \* Scale) + H / 2));

}

else

{

g.DrawLine(Pens.Blue, Convert.ToSingle(C2 / A2 \* Scale + W / 2), 0, Convert.ToSingle(C2 / A2 \* Scale + W / 2), H);

}

if ((A1==0 && A2==0 && C1/B1==C2/B2) || (B1==0 && B2==0 && C1/A1==C2/A2) || (B1!=0 && B2!=0 && A1/B1==A2/B2 && C1/B1==C2/B2))

g.DrawString("Графики накладываеются", f, br, 15, 15);

else if ((A1==0 && A2==0 && C1/B1!=C2/B2) || (B1==0 && B2==0 && C1/A1!=C2/A2) || (B1!=0 && B2!=0 && A1/B1==A2/B2 && C1/B1!=C2/B2))

g.DrawString("Графики параллельны", f, br, 15, 15);

else

{

double x, y;

if (A1==0 && B2==0)

{

x = C2 / A2;

y = C1 / B1;

}

else if (A2 == 0 && B1 == 0)

{

x = C1 / A1;

y = C2 / B2;

}

else

{

y = (C1 / A1 - C2 / A2) / (B1 / A1 - B2 / A2);

x = C1 / A1 - B1 \* y / A1;

}

g.DrawString("Точка пересечения [" + Convert.ToString(x) + ";" + Convert.ToString(y) + "]", f, br, 15, 15);

g.FillEllipse(Brushes.Yellow,

Convert.ToSingle(W / 2 + x \* Scale - 3), Convert.ToSingle(H / 2 - y \* Scale - 3), 6, 6);

g.DrawEllipse(Pens.Black,

Convert.ToSingle(W / 2 + x \* Scale - 3), Convert.ToSingle(H / 2 - y \* Scale - 3), 6, 6);

}

}

}

}

Menu\_form.cs

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;

namespace Kursovaya

{

public partial class Menu\_form : Form

{

object tl;

public Menu\_form(object sender)

{

InitializeComponent();

tl = sender;

}

private void TL\_menuItem\_Click(object sender, EventArgs e)

{

(tl as Main\_form).Visible = true;

this.Dispose();

}

private void Solve\_menuItem\_Click(object sender, EventArgs e)

{

Form1 fr = new Form1();

fr.Show();

}

private void Desc\_menuItem\_Click(object sender, EventArgs e)

{

Description\_form fr = new Description\_form();

fr.Show();

}

private void FC\_menuItem\_Click(object sender, EventArgs e)

{

Flowchart\_form fr = new Flowchart\_form();

fr.Show();

}

private void Exit\_menuItem\_Click(object sender, EventArgs e)

{

Application.Exit();

}

private void Ref\_menuItem\_Click(object sender, EventArgs e)

{

ref\_form fr = new ref\_form();

fr.Show();

}

private void About\_nemuItem\_Click(object sender, EventArgs e)

{

About\_form fr = new About\_form();

fr.Show();

}

private void Menu\_form\_FormClosed(object sender, FormClosedEventArgs e)

{

Application.Exit();

}

}

}

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Kursovaya

{

static class Program

{

/// <summary>

/// Главная точка входа для приложения.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Main\_form());

}

}

}

Main\_form.cs

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;

namespace Kursovaya

{

public partial class Main\_form : Form

{

public Main\_form()

{

InitializeComponent();

}

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

{

Menu\_form mf = new Menu\_form(this);

mf.Show();

this.Visible = false;

}

}

}