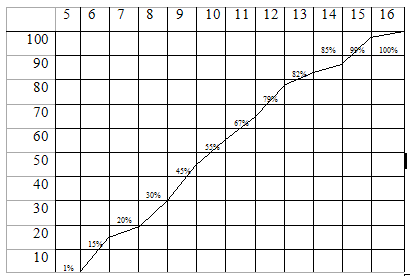
ФГБОУ ВПО

«Уфимский государственный авиационный технический университет»

**Кафедра Информатики**



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

## ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

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

|  |
| --- |
| 1308.301000.000ПЗ |
| (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](#_Toc417952529)

[1 Введение 3](#_Toc417952530)

[1.1 Класс задач 3](#_Toc417952531)

[1.2 Пример задачи 5](#_Toc417952532)

[2 Описание метода 7](#_Toc417952533)

[2.1 Метод Гаусса 7](#_Toc417952534)

[2.2 Блок-схема 7](#_Toc417952535)

[2.3 Код программы 7](#_Toc417952536)

# Введение

## Класс задач

Задачи, связанные с анализом работы электрических цепей, могут формулироваться разными способами, например:

* расчет токов в системе;
* нахождение электродвижущей силы (ЭДС);
* вычисление внутренних сопротивлений источников ЭДС;
* расчет значений сопротивления каждой ветки электрической цепи.

Математическая модель для такого рода задач создается на основе законов Кирхгофа.

В изолированной замкнутой электрической цепи выполняются следующие законы Кирхгофа:

1. Алгебраическая (с учетом знаков) сумма входящих и выходящих токов для любого узла контура равна нулю, т.е. сколько тока «втекает» в узел по одним ветвям (эти токи берутся со знаком «+»), столько из него «вытекает» по другим (эти токи берутся со знаком «–»). Математически это можно выразить следующим образом:

 (1.1)

где N – общее количество узлов цепи, k – номер узла.   
Узлом называется такая точка электрической схемы, в которой сходятся как минимум три тока. Все элементы электрической цепи условно можно разделить на активные и пассивные. Активным называется элемент, содержащий в своей структуре источник электрической энергии. К пассивным относятся элементы, в которых рассеивается (резисторы) или накапливается (катушка индуктивности и конденсаторы) энергия.

1. Алгебраическая сумма падений напряжения на элементах в любом контуре равна нулю, то есть алгебраическая сумма ЭДС в контуре равна алгебраической сумме падений напряжений на пассивных элементах этого контура.   
   При суммировании ЭДС падения напряжений берутся со знаком «+», если их направления совпадают с произвольно выбранным направлением обхода контура и со знаком «-» в противном случае.   
   Считается, что если при выбранном положительном направлении обхода контура первым встречается положительный полюс источника, то ЭДС будет отрицательной, если первым встречается отрицательный, то ЭДС будет положительной. Математически это можно записать следующим образом:

 (1.2)

где NR и NE – число пассивных элементов и ЭДС в контуре соответственно.

Замечание. Если в схеме присутствуют реальные источники тока, то их, при помощи эквивалентных преобразований, следует заменить на идеальные. В случае с источниками постоянного тока, отличие реальных от идеальных заключается в том, что реальные источники тока обладают внутренним сопротивлением, которое обозначается как . Следовательно, математическое представление второго закона Кирхгофа можно заменить на следующее:

 (1.3)

Эти уравнения применяются только к одному контуру. Между тем, вся электрическая схема может быть разбита на несколько контуров. Для каждого из замкнутых контуров, которые можно мысленно выделить в разветвленной цепи, можно составить приведенные выше уравнения, однако, независимыми будут только уравнения для тех контуров, которые нельзя получить наложением других контуров друг на друга.

## Пример задачи

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 1.2.1 Многоконтурная схема электрической цепи к задаче.

Для электрической цепи, изображенной на Рисунок 1.2.1, закон Кирхгофа для токов узле b:

i1 = i2 + i3.

Закон Кирхгофа применительно к левому и внешнему контурам цепи:



Выразив напряжения через токи и сопротивления, получим:



Таким образом, получается система трех уравнений с тремя неизвестными i1, i2 и i3. Ее можно записать так, чтобы в правых частях уравнений находились неизвестные, умноженные на соответствующие коэффициенты, а в левых – свободные члены:

 (2.1)

Матрица коэффициентов и вектор правых частей системы запишутся так:



Это и есть математическая модель этой задачи. Таким образом, ее решение сводится к решению системы линейных уравнений относительно i1, i2, i3.

# 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.

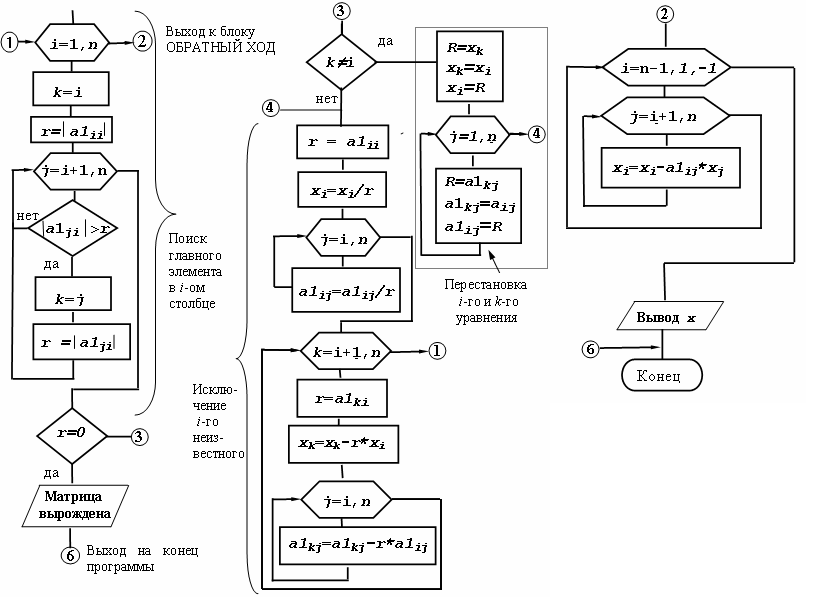
В методе Гаусса происходит деление строк на соответствующие ведущие элементы, поэтому, если на каком-то *k*-м шаге на главной диагонали окажется нулевой элемент  то среди элементов  (i=k+1,…, n) следует найти ненулевой и перестановкой строк переместить его на главную диагональ, а затем продолжить вычисления.

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

Программа для решения СЛАУ методом Гаусса была написана на языке программирования C# в IDE 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. Form2 – 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.

# 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).

Разработанная программа успешно решает СЛАУ от 1-го до 10-го порядка, имеет возможность визуализации решения СЛАУ 2-го порядка, а так же имеет встроенную справку по методу Гаусса, и его блок-схему, а так же справку по самой программе.

# 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
3. <https://msdn.microsoft.com/ru-ru/library/> Каталог API (Microsoft) и справочных материалов
4. Еще что нибудь

# 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");

}

}

}

Form2.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)

{

(tl as Main\_form).Dispose();

}

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)

{

(tl as Main\_form).Dispose();

}

}

}

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;

}

}

}