МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«Київський політехнічний інститут»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра СКС

Лабораторна робота №3

з дисципліни

Алгоритми та методи обчислень

на тему:

**«Розв’язання систем алгебраїчних лінійних рівнянь»**

Виконав студент Перевірив:

Групи КВ-64 \_\_\_\_\_\_\_\_\_\_\_

### Подольський Сергій Валентинович

залікова книжка № КВ6415 \_\_\_\_\_\_\_\_\_\_\_\_\_(бали)

#### VІ семестр

Київ-2009

1. **Постановка задачі.**

Розв’язати систему лінійних алгебраїчних рівнянь, задану за варіантом, методом виключення Гауса-Жордана та методом простої ітерації:

2x[0] + 11x[1] + 13x[2] + 15x[3] = 33

12x[0] + 38x[1] + 18x[2] + 7x[3] = 114

20x[0] + x[1] + 27x[2] + 5x[3] = 3

15x[0] + 2x[1] + 15x[2] + 5x[3] = 6

1. **Текст програми.**

Програма реалізована в середовищі розробки Microsoft Visual Studio 2008 на платформі .NET 3.5 мовою програмування C#.

using System;

using System.Data;

using System.Linq;

using System.Windows.Forms;

namespace NumLab3

{

public partial class MainClass : Form

{

/// <summary>

/// Решение системы линейных уравнений методом Гаусса-Жордана

/// </summary>

/// <param name="matrix">Исходная таблица коефициентов и свободных членов</param>

/// <returns>Вектор корней</returns>

double[] GaussGordan(double[,] matrix)

{

int size = table.Rows.Count;

for (int k = 0; k < size; k++)

{

double current = matrix[k, k];

// Деление строки

for (int j = 0; j < size + 1; j++)

matrix[k, j] /= current;

// Отнимание строк

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

if (i != k)

{

double a = matrix[i, k];

for (int j = k; j < size + 1; j++)

matrix[i, j] -= matrix[k, j] \* a;

}

}

// Вернуть результат - вектор корней

double[] result = new double[size];

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

result[i] = matrix[i, size];

return result;

}

/// <summary>

/// Решение системы линейных уравнений методом Гаусса-Жордана

/// </summary>

/// <param name="matrix">Исходная таблица коефициентов и свободных членов</param>

double[] DirectIteration(double[,] matrix)

{

int k = 0, size = table.Rows.Count; // Номер итерации; количество неизвестных

double eps = Math.Pow(10, (double)numericUpDownEps.Value); // Точность

double[] x0, x = new double[size]; // Массив для хранения прошлых приближений, затем погрешностей

for (int i = 0; i < size; i++) // Начальные приближения - свободные члены

x[i] = matrix[i, size];

do

{

x0 = (double[])x.Clone();

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

{

double sum = 0;

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

if (i != j)

sum += matrix[i, j] \* x0[j];

x[i] = (matrix[i, size] - sum) / matrix[i, i];

}

// Присваиваиваем элементам массива прошлых корней значения погрешностей

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

x0[i] = Math.Abs(x0[i] - x[i]);

k++;

if (k > 1000) throw new Exception("Too much iterations");

}

while (x0.Max() > eps);

listBox.Items.Add("Total iterations: " + k.ToString());

return x; // Вернуть результат - вектор корней

}

/// <summary>

/// Таблица исходных пользовательских данных

/// </summary>

DataTable table = new DataTable("dataTable");

/// <summary>

/// Конструктор формы

/// </summary>

public MainClass()

{

InitializeComponent();

dataGridView.DataSource = table;

// Последняя колонка таблицы (свободный член)

DataColumn last = new DataColumn("=", typeof(double));

last.AllowDBNull = false;

last.DefaultValue = 0;

table.Columns.Add(last);

numericUpDown\_ValueChanged(null, null);

radioButtonGaussJordan\_CheckedChanged(null, null);

}

/// <summary>

/// Изменение количества неизвестных

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void numericUpDown\_ValueChanged(object sender, EventArgs e)

{

DataColumn last = table.Columns[table.Columns.Count - 1];

table.Columns.Remove(last);

for (int i = (int)numericUpDownSize.Value; i < dataGridView.Rows.Count; i++)

{

table.Rows.RemoveAt(i);

table.Columns.RemoveAt(i);

}

for (int i = dataGridView.Rows.Count; i < (int)numericUpDownSize.Value; i++)

{

DataColumn column = new DataColumn(string.Format("X[{0}]", i), typeof(double));

column.AllowDBNull = false;

column.DefaultValue = 1;

table.Columns.Add(column);

table.Rows.Add(table.NewRow());

}

table.Columns.Add(last);

}

/// <summary>

/// Нажатие кнопки "Solve"

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

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

{

try

{

listBox.Items.Clear();

double[,] matrix = new double[table.Rows.Count, table.Columns.Count];

for (int i = 0; i < table.Rows.Count; i++)

for (int j = 0; j < table.Columns.Count; j++)

matrix[i, j] = (double)table.Rows[i][j];

double[] result;

if (radioButtonGaussJordan.Checked)

result = GaussGordan(matrix);

else

result = DirectIteration(matrix);

for (int i = 0; i < result.Length; i++)

listBox.Items.Add(string.Format("X[{0}] = {1}", i, result[i]));

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

/// <summary>

/// Загрузка таблицы

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

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

{

try

{

if (openFileDialog.ShowDialog() == DialogResult.OK)

{

DataTable loaded = new DataTable();

loaded.ReadXml(openFileDialog.FileName);

numericUpDownSize.Value = loaded.Rows.Count;

dataGridView.DataSource = table = loaded;

}

}

catch (Exception ex)

{

MessageBox.Show(ex.Message, "File error", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

/// <summary>

/// Сохранение таблицы

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

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

{

if (saveFileDialog.ShowDialog() == DialogResult.OK)

table.WriteXml(saveFileDialog.FileName, XmlWriteMode.WriteSchema, true);

}

/// <summary>

/// Доступность элементов

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void radioButtonGaussJordan\_CheckedChanged(object sender, EventArgs e)

{

labelEps.Enabled = numericUpDownEps.Enabled = radioButtonDirectIteration.Checked;

}

/// <summary>

/// Обработка ошибок

/// </summary>

/// <param name="sender"></param>

/// <param name="e"></param>

private void dataGridView\_DataError(object sender, DataGridViewDataErrorEventArgs e)

{

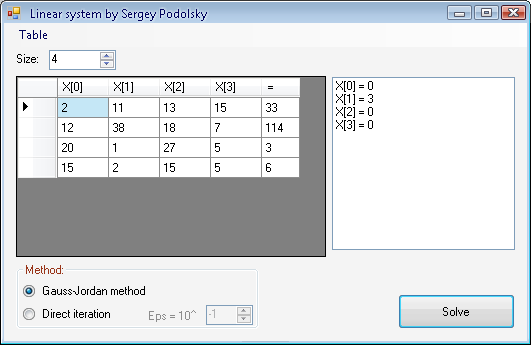
e.Cancel = MessageBox.Show("Invalid value", "Error", MessageBoxButtons.OKCancel, MessageBoxIcon.Warning) == DialogResult.OK;

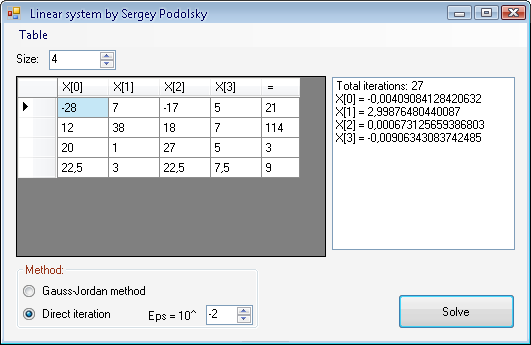
}

}

}

**3. Результати роботи програми.**

****

****

**4. Висновки.**

При розв’язанні заданої за варіантом системи рівнянь за допомогою методу простої ітерації вихідна система заздалегідь була перетворена до придатного до ітерацій вигляду і збережена в файл для можливості подальшого завантаження.

Вихідна матриця була перетворена наступним чином:

2x[0] + 11x[1] + 13x[2] + 15x[3] = 33 (1)

12x[0] + 38x[1] + 18x[2] + 7x[3] = 114 (2)

20x[0] + x[1] + 27x[2] + 5x[3] = 3 (3)

15x[0] + 2x[1] + 15x[2] + 5x[3] = 6 (4)

-18x[0] + 7x[1] –17x[2] + 5x[3] = 21 (1) – (4) \* 2

12x[0] + 38x[1] + 18x[2] + 7x[3] = 114 (2)

20x[0] + x[1] + 27x[2] + 5x[3] = 3 (3)

22,5x[0] + 3x[1] + 22,5x[2] + 7,5x[3] = 9 (4) \* 1,5

Використана література

1. Демідович, Маран – «Основи обчислювальної математики».
2. Каліткін – «Чисельні методи».
3. Хеммінг – «Чисельні методи оптимізації».
4. MSDN 2008.
5. Карлі Ватсон – С#.NET.