МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ОРЛОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИМЕНИ И.С. ТУРГЕНЕВА»

Кафедра информационной безопасности

**Отчет**

по лабораторной работе №7

на тему: «**Решение на ЭВМ систем линейных булевых уравнений методом Гауса**»

по дисциплине «Информационная безопасность»

Выполнили: Кожухова О.А. Шифр: 170582

Шорин В.Д. Шифр: 171406

Институт приборостроения, автоматизации и информационных технологий

Направление: 09.03.04 «Программная инженерия»

Группа: 71-ПГ

Проверил: Еременко В.Т.

Отметка о зачете:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата «\_\_\_\_» \_\_\_\_\_\_ 2021г.

Орел, 2021 г.

**Задание**

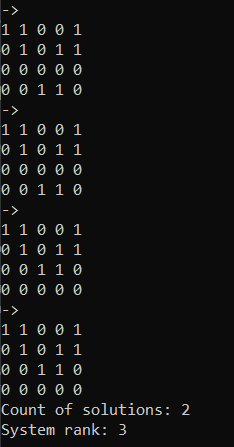
Запрограммировать алгоритм Гаусса для решения СЛБУ с квадратной (n х n)-матрицей коэффициентов А, n ≤ 30. При этом входными данными программы являются число n ≤ 30 и имя файла, в котором записана расширенная матрица СЛБУ, то есть матрица с коэффициентами и правыми частями. Файл должен иметь текстовый формат, матрица записывается по строкам слитно, без каких-либо разделительных символов между строками и между элементами строки.

Программа должна выполнять следующие ***функции:***

- определять совместность СЛБУ;

- вычислять ранг СЛБУ;

**Ход работы**



**Код**

**«Program.cs»**

using System;

namespace IS\_L\_7

{

class Program

{

//private static byte[,] Matrix = new byte[4, 5]

//{

// { 0, 1, 0, 0, 1 },

// { 1, 1, 0, 0, 1 },

// { 0, 0, 1, 0, 0 },

// { 1, 1, 0, 0, 1 },

//};

private static byte[,] Matrix = new byte[4, 5]

{

{ 1, 1, 0, 0, 1 },

{ 0, 1, 0, 1, 1 },

{ 1, 1, 0, 0, 1 },

{ 0, 0, 1, 1, 0 },

};

static void Main(string[] args)

{

//Console.Write("Enter n: ");

//int n = Convert.ToInt32(Console.ReadLine());

int n = 4;

if ( n > 30 || n < 1)

{

Console.WriteLine("n должно быть больше 0 и меньше или равно 30");

return;

}

bool[] A = new bool[n \* n];

bool[] B = new bool[n];

bool[] X = new bool[n];

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

{

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

{

A[i \* n + j] = (Matrix[i, j] != 0);

}

B[i] = (Matrix[i, 4] != 0);

}

int systemRank;

int solutionsCount = GaussianEliminationAlgorithm(A, B, X, n, out systemRank);

Console.WriteLine("Count of solutions: " + solutionsCount);

if (solutionsCount == 1)

{

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

Console.Write(X[i] + " ");

Console.WriteLine();

}

Console.WriteLine("System rank: " + systemRank);

}

private static int GaussianEliminationAlgorithm(bool[] A, bool[] B, bool[] X, int n, out int systemRank)

{

int i\_swap, j = 0;

bool tmp;

while (!IsTriangleMatrix(A, n) && j < n)

{

i\_swap = -1;

for (int i = j; i < n; i++)

{

if (A[i \* n + j])

{

i\_swap = i;

break;

}

}

if (i\_swap >= 0)

{

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

{

tmp = A[j \* n + k];

A[j \* n + k] = A[i\_swap \* n + k];

A[i\_swap \* n + k] = tmp;

}

tmp = B[j];

B[j] = B[i\_swap];

B[i\_swap] = tmp;

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

{

if (A[k \* n + j])

{

for (int l = j; l < n; l++)

{

A[k \* n + l] ^= A[j \* n + l];

}

B[k] ^= B[j];

}

}

}

Console.WriteLine("->");

PrintA0(A, B, n);

j++;

}

int zeroRows = 0;

bool inconsistent = false;

bool haveTrue;

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

{

haveTrue = false;

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

{

haveTrue |= A[i \* n + j];

if (haveTrue) break;

}

if (!haveTrue)

{

zeroRows++;

if (B[i]) inconsistent = true;

}

}

systemRank = n - zeroRows;

if (inconsistent) return 0;

if (zeroRows > 0) return Convert.ToInt32(Math.Pow(2, zeroRows));

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

{

X[i] = B[i];

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

X[i] ^= A[i \* n + j] && X[j];

}

return 1;

}

private static bool IsTriangleMatrix(bool[] A, int n)

{

bool flag = true;

for (int i = 1; i < n; i++)

{

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

{

if (A[i \* n + j])

{

flag = false;

break;

}

}

if (!flag)

break;

}

return flag;

}

private static void PrintA0(bool[] A, bool[] B, int n)

{

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

{

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

{

Console.Write((A[i \* n + j] == false) ? "0" + " " : "1" + " ");

}

Console.WriteLine((B[i] == false) ? "0" : "1");

}

}

}

}