Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное   
учреждение высшего образования

Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского

Институт информационных технологий, математики и механики

**Отчет по лабораторной работе**

**«Метод Гаусса»**

**Выполнила**:

студентка группы 3821Б1ПМ2

Анисимова Ю.Д.

**Проверил**:

преподаватель каф. МОСТ,

Волокитин В.Д.

Нижний Новгород

2022

**Содержание**

Постановка задачи 3

Метод решения 4

Руководство пользователя 6

Описание программной реализации 8

Подтверждение корректности 9

Результаты экспериментов 10

Заключение 13

Приложение 14

# Постановка задачи

Задачей моей лабораторной работы является реализация метода Гаусса с выбором ведущего элемента, используя реализованный самостоятельно шаблонный класс вектор, а также класс квадратная матрица, являющийся шаблоном класса вектор от вектора. Метод Гаусса должен принимать на вход систему линейных алгебраических уравнений, а выводить на экран вектор значений. Нужно описать реализацию и алгоритмы работы программы. Необходимо экспериментально подтвердить корректность работы сознанного алгоритма.

# Метод решения

Основным алгоритмом программы является метод Гаусса – метод решения системы линейных алгебраических уравнений, основанный на исключении переменных путём элементарных преобразований.

Стандартный метод Гаусса состоит из 2 этапов:

*Прямой ход:*

Исключения Гаусса основаны на идее последовательного исключения переменных по одной до тех пор, пока не останется только одно уравнение с одной переменной в левой части. Затем это уравнение решается относительно единственной переменной. Таким образом, систему уравнений приводят к треугольной (ступенчатой) форме. Для этого среди элементов первого столбца матрицы выбирают максимальный элемент и перемещают его на крайнее верхнее положение перестановкой строк. Затем нормируют первое уравнение, разделив его на коэффициент ai1, где i– номер столбца. Затем из остальных строк вычитают получившуюся после перестановки первую строку, умноженную на элемент a1j, где j – номер строки. Получают новую систему уравнений, в которой заменены соответствующие коэффициенты. После того, как указанные преобразования были совершены, первую строку и первый столбец мысленно вычёркивают и продолжают указанный процесс для всех последующих уравнений пока не останется уравнение с одной неизвестной.

*Обратный ход:*

Обратная подстановка предполагает подстановку полученного на предыдущем шаге значения переменной xn в предыдущие уравнения. Эта процедура повторяется для всех оставшихся решений.

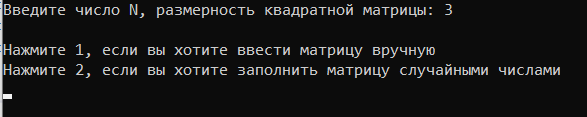
В моей работе приведён метод Гаусса с выбором ведущего элемента. Он неотличим от стандартного алгоритма за тем исключением, что на каждой итерации в качестве ведущего элемента выбирается не просто следующий ненулевой элемент, а наибольший по модулю в столбце. Выбор наибольшего по модулю элемента значительно снижает потерю точности при делении и последующих действиях.

# Руководство пользователя

После запуска программы появляется окно ввода, где предлагается ввести одно натуральное число, которое задает размерность квадратной матрицы.

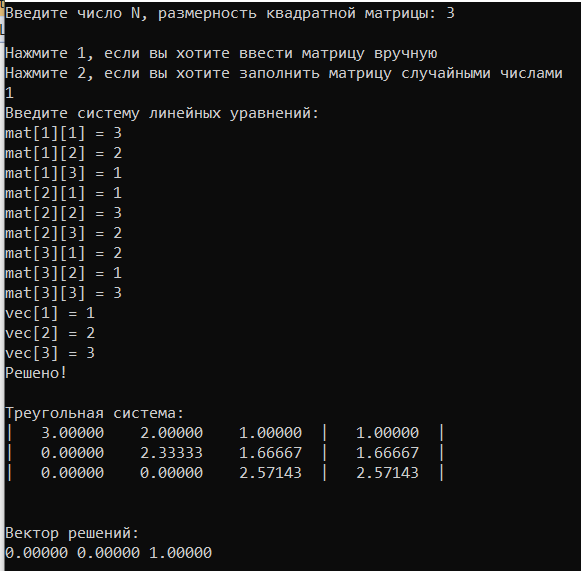


Далее предлагается выбрать:

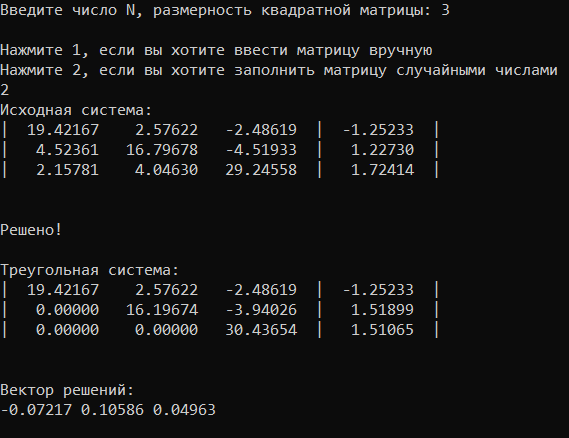
* Заполнить матрицу вручную (для этого необходимо нажать 1)
* Заполнить матрицу случайными числами (для этого необходимо нажать 2)

Если пользователь нажал 1, то предлагается ввести все элементы матрицы вручную. После этого выходит сообщение «Решено!», если матрица решена без ошибок, в противном случае выводится сообщение «Решено с ошибкой!».

Далее все элементы матрицы записываются в треугольную систему и выводится вектор решений.

Если же система не имеет решений выходит сообщение о том, что система несовместна.

Если при выборе способа ввода матрицы, пользователь нажал 2, матрица заполняется случайными числами и также элементы матрицы записываются в треугольную систему и выводится вектор решений.



# Описание программной реализации

В программе содержится 3 файла:

* ClassVector (в нем находится класс вектор)
* Matrix1 (в нем находится класс матрица и класс Gauss, где реализован метод Гаусса)
* Laba3 (в нем находится int main, где реализовано меню программы)

**Vector** – шаблонный класс вектор, хранит значения любого типа, при вызове создает динамический массив. Имеет в себе различные методы, функции и перегруженные бинарные операции «+», «-», «\*», «/» и «=».

**Matrix** – шаблонный класс матрица, является наследником класса вектор от вектора.

**Gauss** – класс, являющийся наследником класса матрица, в этом классе реализован метод Гаусса.

# Подтверждение корректности

Для подтверждения корректности в программе присутствуют операторы исключений (throw). Если у системы нет ни одного решения, то выводится сообщение «Система несовместна», а если при решении происходит деление на 0, то выводится сообщение «Деление на 0». Также если система решена без ошибок выходит сообщение «Решено!», в противном случае «Решено с ошибкой!».

# Результаты экспериментов

Тестирование производилось на 6 СЛУ

Часть проверенных решений прикреплено ниже.

|  |  |  |
| --- | --- | --- |
| **СЛУ** | **Решение вручную** | **Решение программы** |
|  | Нет решений (система несовместна) |  |
|  | (1; 2; 1) |  |
|  | (-2; 2; -3; 3) |  |
|  | Нет решений (система несовместна) |  |
|  | (2; -3; -1) |  |
|  | Бесконечно много решений |  |

СЛУ были решены вручную и с помощью онлайн калькулятора метода Гаусса, затем производились решение через программу и сравнение результатов.

На всех тестах программа отработала корректно.

Для проверки использовался онлайн-калькулятор:

https://matworld.ru/calculator/gauss-method-online.php

# Заключение

В ходе лабораторной работы я реализовала метод Гаусса с выбором ведущего элемента, используя реализованный самостоятельно шаблонный класс вектор, а также класс квадратная матрица, являющийся шаблоном класса вектор от вектора. Метод Гаусса принимает на вход систему линейных алгебраических уравнений, а выводит на экран вектор значений. Описала реализацию и алгоритмы работы программы. Также экспериментально подтвердила корректность работы сознанного алгоритма.

# Приложение

class Matrix : public Vector<Vector<T>> {

public:

Matrix(size\_t stolb, const Vector<T>& stroka) : Vector<Vector<T>>(stolb, stroka) {}

const Vector<T> operator \* (Vector<T>& other)

{

Vector<T> res(this->size);

for (int i = 0; i < this->size; i++)

{

res[i] = this->mas[i] \* other;

}

return res;

}

};

class Gauss {

private:

size\_t size;

Matrix<double> mat;

Vector<double> vec;

Vector<double> res;

public:

Gauss(size\_t stolb, const Vector<double>& stroka) : mat(stolb, stroka), vec(stolb), res(stolb)

{

this->size = stolb;

}

size\_t GetSize() { return size; }

Vector<double>& Solve()

{

Matrix<double> matCopy(mat);

Vector<double> vecCopy(vec);

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

{

double k;

int maxJ = j;

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

{

if (abs(mat[maxJ][j]) < abs(mat[i][j]))

{

maxJ = i;

}

}

if (j != maxJ)

{

swap(mat[j], mat[maxJ]);

swap(mat[j], mat[maxJ]);

}

if (mat[maxJ][j] == 0)

{

throw exception("Система несовместна");

}

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

{

if (mat[j][j] != 0)

{

k = mat[i][j] / mat[j][j];

}

else

{

throw exception("деление на 0");

}

mat[i] -= mat[j] \* k;

vec[i] -= vec[j] \* k;

}

}

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

{

double summ = 0;

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

{

summ += mat[i][j] \* res[j];

}

res[i] = (vec[i] - summ) / mat[i][i];

}

if (examination(matCopy, vecCopy))

{

cout << "Решено!\n" << endl;

}

else

{

cout << "Решено с ошибкой!\n" << endl;

}

return res;

}

bool examination(Matrix<double>& Mat, Vector<double>& Vec)

{

Vector<double> exam = Mat \* res;

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

{

if (abs(exam[i] - Vec[i]) >= 0.0000000001)

{

return false;

}

}

return true;

}

void random()

{

double low = -5.0;

double max = 5.0;

srand(static\_cast <unsigned> (time(0)));

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

{

vec[i] = low + ((double)(rand()) / ((double)(RAND\_MAX / (max - low))));

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

{

mat[i][j] = low + ((double)(rand()) / ((double)(RAND\_MAX / (max - low))));

if (i == j)

mat[i][j] = size \* max + ((double)(rand()) / ((double)(RAND\_MAX / (size \* max))));

}

}

}

friend ostream& operator << (ostream& out, Gauss& system);

friend istream& operator >> (istream& in, Gauss& system);

};

ostream& operator << (ostream& out, Gauss& system)

{

cout << std::fixed;

for (int i = 0; i < system.GetSize(); i++)

{

cout << "|";

for (int j = 0; j < system.GetSize(); j++)

{

cout.width(10);

cout << std::setprecision(5)

<< system.mat[i][j] << ' ';

}

cout << " |";

cout.width(10);

cout << std::setprecision(5)

<< system.vec[i] << " |" << '\n';

}

cout << "\n\n";

return out;

}

istream& operator >> (istream& in, Gauss& system)

{

for (int i = 0; i < system.GetSize(); i++)

{

for (int j = 0; j < system.GetSize(); j++)

{

cout << "mat[" << i + 1 << "][" << j + 1 << "] = ";

cin >> system.mat[i][j];

}

}

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

{

cout << "vec[" << i + 1 << "] = ";

cin >> system.vec[i];

}

return in;

}