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»**

**Выполнил**:

студент группы 3821Б1ПМ2

Деревянкин К. Е.

**Проверил**:

преподаватель каф. МОСТ,

Волокитин В.Д.

Нижний Новгород

2021

**Содержание**

[Постановка задачи 2](#_Toc1818946334)

[Метод решения 3](#_Toc1858084947)

[Руководство пользователя 4](#_Toc1054998367)

[Описание программной реализации 5](#_Toc2020843715)

[Подтверждение корректности 6](#_Toc1536657880)

[Заключение 7](#_Toc1972571477)

[Приложение 8](#_Toc2027120728)

# Постановка задачи

Задачей лабораторной работы была реализация метода Гаусса с прямым и обратным ходом на языке программирования C++.

# Метод решения

Метод Гаусса – последовательное исключение неизвестных. Этот метод используется для решения квадратных систем линейных алгебраических уравнений. Если СЛАУ имеет одно или несколько решений, то система называется совместимой, в другом случае - несовместимой.

В данной лабораторной работе я использовал метод Гаусса с ведущим элементом. Его смысл заключается в том, чтобы преобразовать расширенную матрицу к верхнетреугольному или нижнетреугольному виду.

На первом шаге прямого хода метода Гаусса выбирается максимальный по модулю элемент в первом столбце. Этот элемент является ведущим. Если ведущий элемент не является элементом a11, то перестановкой строк помещаем его в a11. При этом соответственно переставляются элементы вектора b. Делим строку на этот элемент, остальные строки делим на элемент, стоящий в том же столбце. Затем происходит тоже самое. То есть в каждом столбце мы выбираем максимальный элемент и эту строчку меняем местами с текущей. При помощи эквивалентных преобразований все элементы ниже становятся 0. Мы получаем верхнетреугольную матрицу.

Далее, когда мы получили ступенчатый вид, применяем обратный ход. Система будет несовместной, если существует строка состоящая из нулей, а элемент из столбца свободных членов соответствующий этой строке не равен 0. В других случаях СЛАУ будет иметь решение.

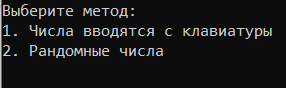
# Руководство пользователя

При запуске программы пользователя просят ввести размер матрицы (рис. 1).



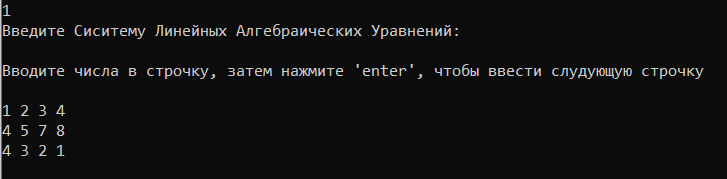
(рис. 1)

Затем пользователя просят выбрать число, где 1 - пользователь вводит числа с клавиатуры, а 2 - вводятся рандомные числа. (рис. 2)



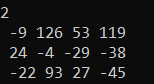
(рис. 2)

Если пользователь выбирает первый вариант, то он вводит числа матрицы и свободные члены в строчку через пробел. А затем нужно нажать “enter”, чтобы перейти на новую строчку. (рис. 3)



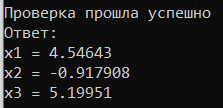
(рис. 3)

Если пользователь выбирает второй вариант, то ему делать ничего не нужно, программа введет числа за него. (рис. 4)



(рис. 4)

Программа выводит ответ и проверку (правильно ли решение или нет). (рис. 5)



(рис. 5)

# Описание программной реализации

void swap(T& vec1, T& vec2) - меняет элементы вектора местами.

Шаблонный класс Vector (template <typename T> class Vector) имеет поля: int size - размер вектора, T\* mas - массив элементов T вектора.

Vector(int size) - конструктор класса выделяет память для массива

~Vector() - деструктор класса, освобождает память, выделенную массивом.

T& GetMas() - возвращает размер вектора size.

Vector& operator=(const Vector& other\_vect) - перегрузка операции присваивания.

Vector operator+ (const Vector& other\_vect) - перегрузка оператора сложение.

Vector operator- (const Vector& other\_vect) - перегрузка оператора разность.

Vector operator\* (const T& scalar) - перегрузка оператора умножение на скаляр.

Vector operator/ (const T& scalar) - перегрузка оператора деление на скаляр.

T& operator [](int i) - перегрузка оператора [].

friend::std::istream& operator>> (std::istream& ist, const Vector& mas) - перегрузка оператора ввод.

friend::std::ostream& operator<< (std::ostream& ost, const Vector& mas) - перегрузка оператора вывод.

Vector operator\*(const Vector& other\_vect) const - перегрузка оператора скалярное произведение.

Шаблонный класс Matrix, который наследуется от Vector<Vector<T>> (template <typename T> class Matrix : public Vector<Vector<T>>)

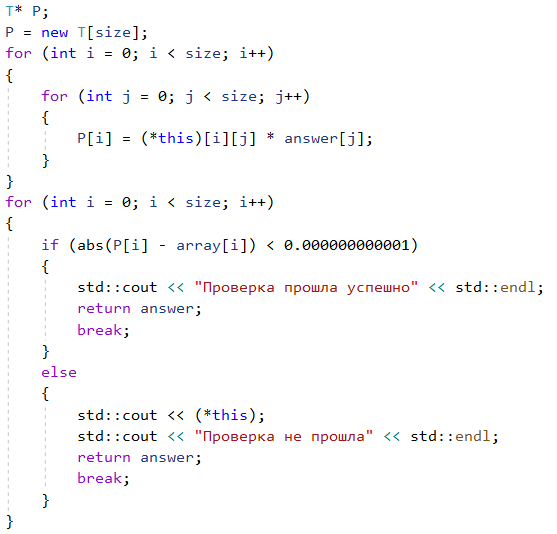
Matrix(int size, const Vector<T>& vectr) : Vector<Vector<T>>(size, vectr)

Шаблонный класс SLAU template <typename T> class SLAU :public Matrix<T> имеет матрицу и столбец свободных членов.

Vector<T> GaussMethod() - решает СЛАУ.

# Подтверждение корректности

Чтобы проверить корректность программы я создал новый массив, который умножил на полученный ответ, то есть должен получить массив свободных членов. И теперь сравниваю полученные данные с исходными. (рис. 6)

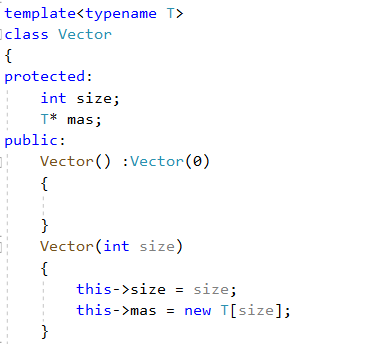


(рис. 6)

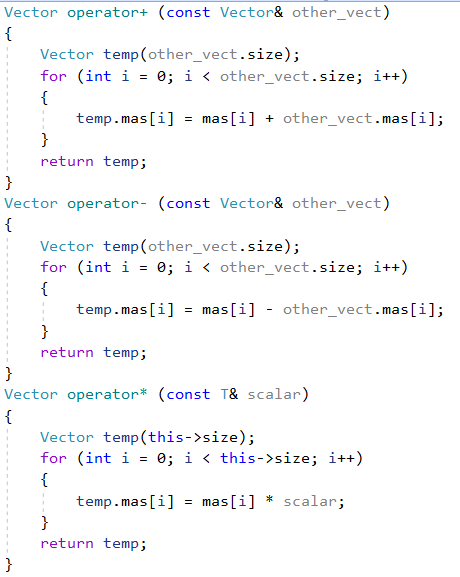
# Заключение

Я реализовал метод Гаусса на языке программировния C++. Создал класс Matrix, отнаследованный от класса Vector, подтвердил корректность программы.

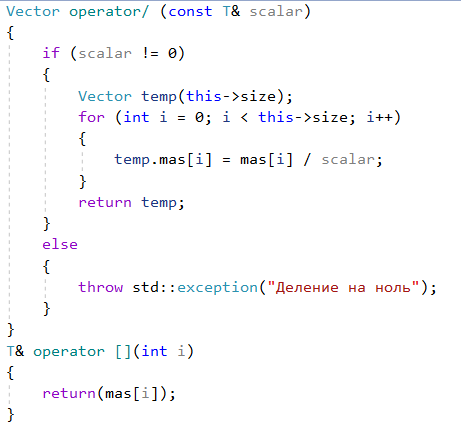
# Приложение



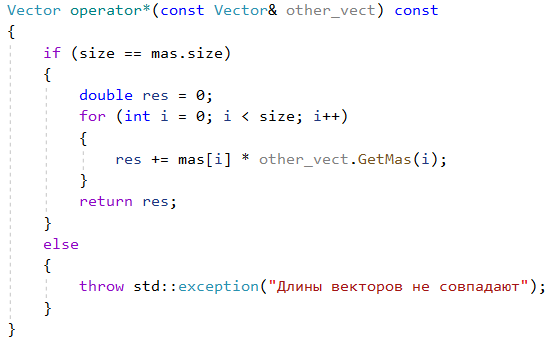
(рис. 7)



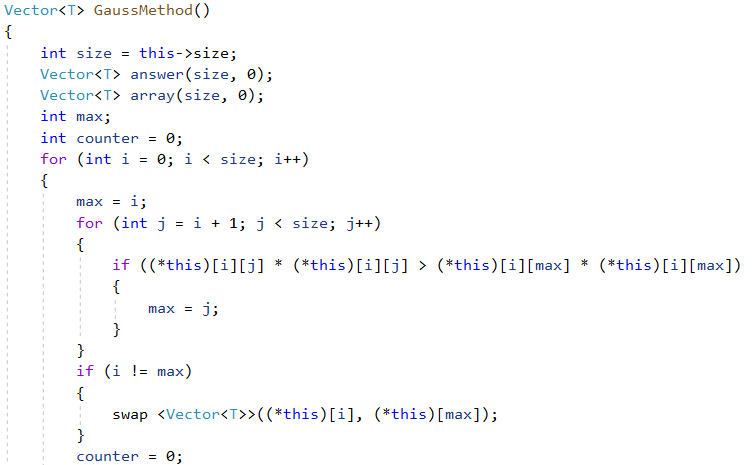
(рис. 8)



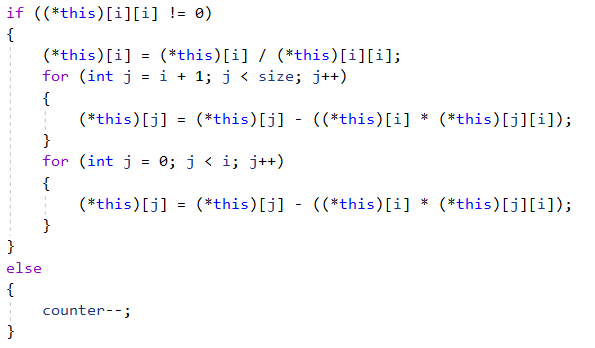
(рис. 9)



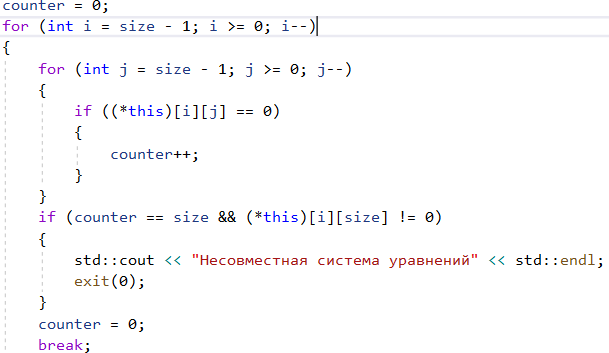
(рис. 10)



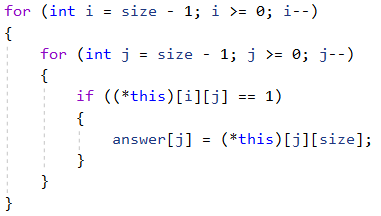
(рис. 11)



(рис. 12)



(рис. 13)



(рис. 14)