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

Фомичев Дмитрий Евгеньевич

**Проверил**:

преподаватель каф. МОСТ,

Волокитин В.Д.

Нижний Новгород

2022

**СОДЕРЖАНИЕ**

[ПОСТАНОВКА ЗАДАЧИ 3](#_Toc103724798)

[МЕТОД РЕШЕНИЯ 4](#_Toc103724799)

[РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 5](#_Toc103724800)

[ОПИСАНИЕ ПРОГРАММНОЙ РЕАЛИЗАЦИИ 7](#_Toc103724801)

[ПОДТВЕРЖДЕНИЕ КОРРЕКТНОСТИ 10](#_Toc103724802)

[ЗАКЛЮЧЕНИЕ 11](#_Toc103724804)

# ПОСТАНОВКА ЗАДАЧИ

― Реализовать классы вектор, матрица и СЛАУ. В последнем реализовать метод Гаусса с выбором ведущего элемента.

― Высчитать погрешность вычисления неизвестных.

― Сделать вывод о проделанной работе.

# МЕТОД РЕШЕНИЯ

1. На первом шаге прямого хода метода Гаусса выбирается максимальный по модулю элемент в первом столбце. Этот элемент является ведущим. Если он равен нулю, то *det*A = 0. Если ведущий элемент не является элементом *а*11, то перестановкой строк помещаем его в  *а*11. При этом соответственно переставляются элементы вектора *b*. Затем применяются формулы метода Гаусса.

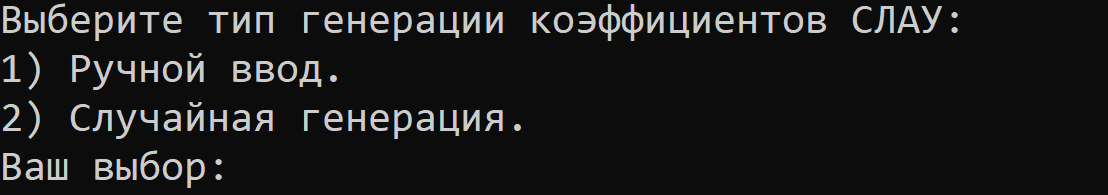
2. На *i*-м шаге прямого хода метода Гаусса непреобразованный столбец – это часть столбца *i*, начиная с элемента *a*ii, то есть *a*ii, *a*i-1i,…,*a*ni. Находим максимальный по модулю элемент  в непреобразованном столбце. Этот элемент является ведущим. Если он равен нулю, то detA = 0. Если ведущий элемент не является элементом *a*ii, то перестановкой строк помещаем его в  *a*ii. При этом соответственно переставляются элементы вектора *b*. Затем применяются формулы метода Гаусса.

3. После (n-1)-го шага получаем верхнюю треугольную матрицу U и преобразованный вектор правой части.  Выполняем обратную подстановку.

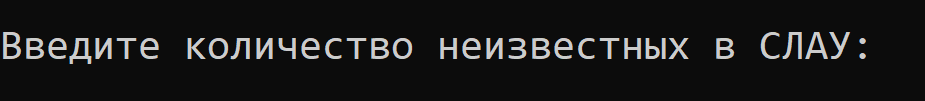
# РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

LAB3.sln

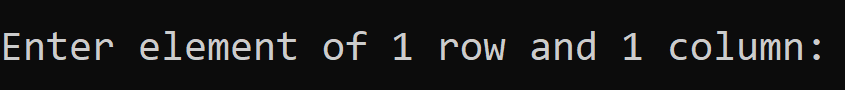
Запустите программу. На экране появится меню выбора типа генерации коэффициентов СЛАУ (рис. 1).

Рис. 1

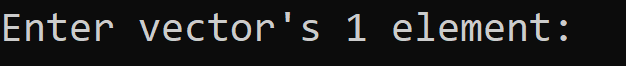
После выбора типа генерации необходимо ввести количество неизвестных в СЛАУ (рис. 2).

Рис. 2

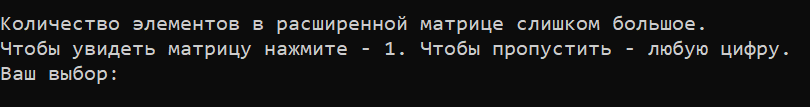
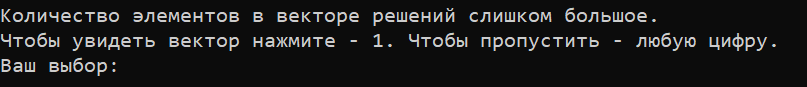
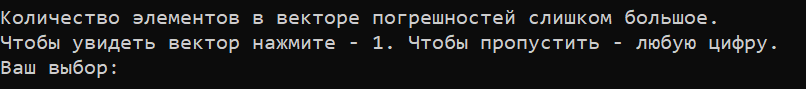
Если вы выбрали ручной ввод коэффициентов, то необходимо ввести каждый коэффициент (рис. 3).

Рис. 3

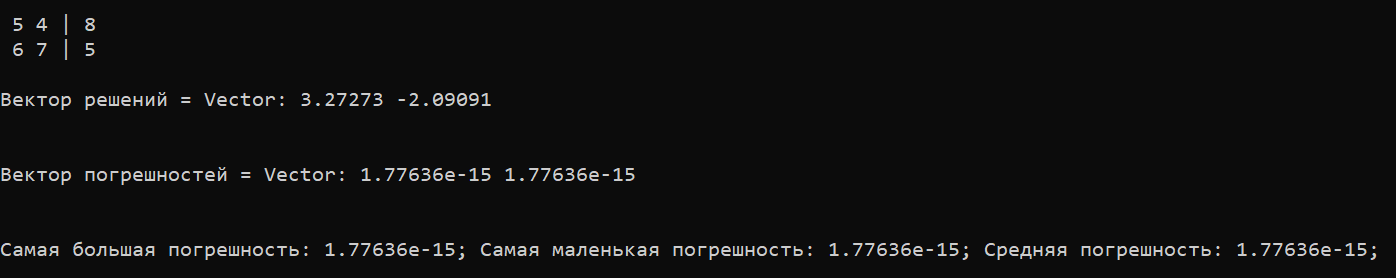
И каждый элемент вектора *b* (рис. 4)

 Рис. 4

Если количество неизвестных в СЛАУ больше 20, то вы можете выбрать – отобразить или нет расширенную матрицу (рис. 5), вектор решений (рис. 6), вектор погрешностей (рис. 7).

Рис. 5Рис. 6Рис. 7

В конце появится: расширенная матрица, вектор решений, вектор погрешностей и информация о максимальной, минимальной и средней погрешности (Рис. 8).

Рис. 8

В случае если последний элемент матрицы после прямого хода метода Гаусса или диагональный элемент в обратном ходе равен нулю, то появится сообщение (Рис. 9).

Рис. 9

Если было введено 0 или отрицательное число при вводе количества неизвестных, то появится сообщение (Рис. 10)

Рис. 10

# ОПИСАНИЕ ПРОГРАММНОЙ РЕАЛИЗАЦИИ

LAB3.1.sln

**CLASS VECTOR**

Шаблонный класс. В protected области реализовано два поля: int size – размер вектора; T\*data – массив, хранящий элементы вектора. В public области реализованы: конструктор, конструктор копирования, деструктор, операторы: [], =, \* вектор на число и число на вектор, / на число, +=, -=; методы: at, resize, set\_vector(задание элементов вектора вручную), zero\_vector(заполнение вектора нулевыми значениями), get\_vector(вывод элементов вектора в консоль), get\_size(получение размера вектора).

**CLASS MATRIX**

Производный класс от class vector<vector<T>> с доступом к полям в protected и методам public vector. В классе реализованы: конструктор, конструктор копирования, методы: set\_matrix(задание элементов матрицы вручную), get\_matrix(вывод элементов матрицы в консоль), at; и оператор [].

**CLASS SLAU**

Шаблонный класс. Содержит private область с полями: int size(размеры матрицы и вектора), matrix<T> A, vector<T> b, vector<T> x, vector<T> error. В public области реализованы: конструктор, методы: get\_SLAU(по выбору выводит на экран расширенную матрицу, вектор решений, вектор погрешностей и всегда информацию о погрешностях) и GAUSS.

Vector<T> GAUSS(matrix<T>& A, vector<T>& b)

Метод Гаусса с выбором ведущего элемента. Создает копии матрицы А и вектора b. В цикле если диагональные элементы меньше установленного epsilon(#define epsilon 0.000001 в текущей версии), то значение заменяется на static\_cast<T>(0). На каждой итерации вычисляется номер максимального элемента в столбце с помощью функции compare(), с помощью swap(vector, vector) меняются местами строчка с максимальным элементом и с текущим диагональным элементом. С помощью swap(T, T) аналогично меняются местами элементы вектора b. В переменную max записывается номер столбца. В следующем цикле вычитаем и последующих строк строку с ведущим элементом. Аналогично для вектора b. Создается vector<T> X, который будет хранить результаты вычислений. Проверяем самый последний элемент на малую величину. В случае если он меньше epsilon, то СЛАУ будет иметь более 2 решений или система несовместна. Иначе делим последнюю строку на ведущий элемент. В следующем цикле в переменную temp записывается сумма temp и произведения элемента вектора решений на элемент матрицы. Если диагональный элемент меньше epsilon, то решений много. Иначе записываем в элемент вектора решений частное от разности элемента вектора b и temp и элемента матрицы.

Int comare(matrix<T>& array, int column\_number)

Функция сравнивает элементы в столбце и возвращает номер наибольшего.

Void random\_matrix(matrix<T>& one, int size)

Заполняет матрицу случайными значениями.

Void random\_vector(vector<T>& one, int size)

Заполняет вектор случайными значениями.

Vector<T> check(matrix& A, vector<T>& x, vector<T>& b)

Вычисляет погрешность вычислений и возвращает в виде вектора error.

Void compare(vector<T>& x, int size, T\* max, T\* min, T\* avg)

Сравнивает элементы вектора error и возвращает наибольшую, наименьшую и среднюю погрешность.

Int main()

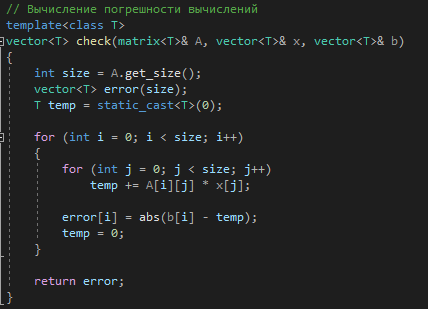
В функции используются функция смены языка setlocale(LC\_ALL, “Russian”) из библиотеки iostream, а также функция menu(). Для последней реализован метод обработки ошибок try\catch. В случае если будет «пойман» throw, выведется сообщение об ошибке и программа будет остановлена.

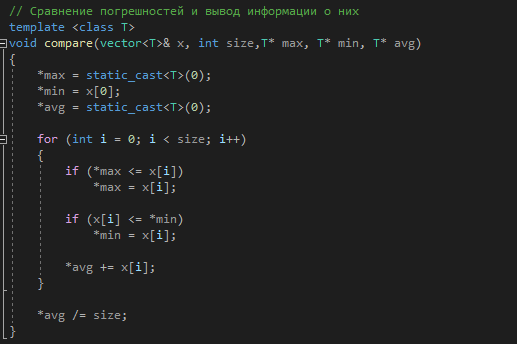
Void menu()

В функции реализовано выбор типа генерации коэффициентов и количества неизвестных в СЛАУ. В случае если введено количество неизвестных <= 0, то программа выдаст ошибку. В switch/case реализован выбор типа генерации коэффициентов.

# ПОДТВЕРЖДЕНИЕ КОРРЕКТНОСТИ

Для подтверждения корректности в программе реализована функция vector<T> check(matrix<T>& A, vector<T>& x, vector<T>& b) (Рис. 11). Она вычитает из вектора b произведение матрицы A и вектора x и записывает значения в вектор error. Для удобства существует функция void compare(vector<T>& x, int size, T\*max, T\*min, T\*avg) (Рис. 12). Она сравнивает все значения вектора error и выводит максимальную погрешность, минимальную и среднюю.

Рис 11.

Рис. 12

# ЗАКЛЮЧЕНИЕ

Я реализовал шаблонный класс vector, от него производный класс matrix и класс SLAU. В классе SLAU реализовал метод Гаусса с выбором ведущего элемента. Высчитал погрешность вычислений данного метода. Сделал отчет о проделанной работе.