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**

**«Реализация метода Гаусса для действительных квадратных матриц с выбором ведущего элемента»**

**Выполнил**:

Студент группы 3824Б1ПМ1

Горев А. С.

**Проверила**:

Бусько П.В.

Нижний Новгород

2025

**Содержание**

[Введение 3](#_Toc199108894)

[Постановка задачи 4](#_Toc199108895)

[Руководство пользователя 5](#_Toc199108896)

[Описание программной реализации 6](#_Toc199108897)

[Результаты экспериментов 12](#_Toc199108898)

[Заключение 15](#_Toc199108899)

[Литература 16](#_Toc199108900)

[Приложение 17](#_Toc199108901)

# Введение

Одной из важнейших задач прикладной математики и инженерных дисциплин является решение систем линейных алгебраических уравнений (СЛАУ), которые широко применяются в физике, экономике, компьютерной графике и других областях. Существует множество численных методов решения СЛАУ, одним из наиболее универсальных и распространённых является метод Гаусса. Он основан на последовательном исключении переменных путём приведения исходной системы к ступенчатому виду и последующего обратного хода.

В рамках данной лабораторной работы ставится задача реализовать метод Гаусса с выбором ведущего элемента по столбцу. Такой подход повышает численную устойчивость метода и позволяет получать корректное решение даже при наличии неудачных начальных условий. В реализации используются средства языка программирования C++, включая механизмы шаблонов и объектно-ориентированного программирования.

Особое внимание в работе уделяется построению обобщённых структур данных: реализуется шаблонный класс вектора, шаблонная матрица как вектор векторов, а также специализированный класс для решения СЛАУ, включающий реализацию метода Гаусса с частичным выбором главного элемента. Программа снабжена обработкой ошибок и демонстрирует универсальность подхода.

# Постановка задачи

В рамках лабораторной работы требуется реализовать метод Гаусса для решения системы линейных алгебраических уравнений с действительной квадратной матрицей и выбором ведущего элемента. Решение должно быть выполнено с использованием языка программирования C++ и включать применение шаблонов и принципов объектно-ориентированного программирования.

Необходимо выполнить следующие требования:

* Реализовать шаблонный класс одномерного вектора;
* Реализовать класс квадратной матрицы как шаблон вектора от вектора;
* Разработать класс системы линейных алгебраических уравнений (СЛАУ), который наследуется от квадратной матрицы и содержит метод Гаусса;
* Метод Гаусса должен принимать в качестве аргумента вектор правой части и возвращать вектор-решение;
* Для повышения удобства и читаемости кода рекомендуется реализовать вспомогательную функцию swap, осуществляющую обмен строк и элементов матрицы.

Программа должна корректно выполнять прямой и обратный ход метода Гаусса с частичным выбором главного элемента, обеспечивать устойчивость вычислений и обрабатывать возможные исключения, связанные с вырожденными или несовместными системами.

# Руководство пользователя

**Описание работы программы.**

Программа реализует метод Гаусса с выбором ведущего элемента для решения систем линейных алгебраических уравнений (СЛАУ) вида: A\*X=B, где:

* A — квадратная матрица коэффициентов размера n × n,
* X — вектор неизвестных,
* B — вектор правых частей.

Алгоритм реализован на языке C++ с применением шаблонных и объектно-ориентированных конструкций. В процессе работы программа запрашивает у пользователя размерность системы, ввод коэффициентов и правой части, после чего производит вычисления и выводит решение.

**Действия пользователя:**

1. Запуск программы.

После запуска на экран выводится приглашение ко вводу размерности матрицы.

1. Ввод размерности.

Введите одно целое число n — размер квадратной матрицы (количество уравнений и переменных).

1. Ввод матрицы коэффициентов.

Программа попросит ввести n строк, каждая из которых содержит n действительных чисел — коэффициенты при переменных в уравнениях.

* + Ввод производится построчно, по одному уравнению за раз.

1. Ввод правой части.

Введите n чисел — правую часть системы (свободные члены).

* + Все значения вводятся в одной строке, через пробел.

1. Получение результата.

После ввода данных программа произведёт вычисления.

* + Если система совместна и имеет единственное решение — оно будет выведено в виде набора значений переменных.
  + Если система вырождена или несовместна — на экран будет выведено сообщение об ошибке.

# Описание программной реализации

**Файл «Lab\_№3.cpp»**

В нем находятся заголовочные файлы (*рис.1*), класс Vector (*Рис. 2)*, класс Matrix (*рис.3*), класс SLAU (*рис.4) и main (рис.5)*.

Изображение выглядит как текст, Шрифт, снимок экрана, типография

Автоматически созданное описание

*Рис. 1 Заголовочные файлы*

Изображение выглядит как текст, снимок экрана

Автоматически созданное описание

*Рис. 2 Класс Vector*

*Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание*

*Рис. 3 Класс Matrix*

*Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание*

*Рис.4 Класс SLAU*

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

*Рис.5 main*

**Подключенные библиотеки:**

* <iostream>, используется для ввода и вывода данных в консольном приложении;
* <vector>, используется для реализации динамических массивов в классах Vector и Matrix;
* <cmath>, предоставляет набор математических функций и констант, которые используются для выполнения различных математических операций;

**Класс Vector:**

* std::vector<T> vect – Внутренний контейнер, хранящий элементы вектора. Использует шаблонный тип T, что позволяет работать с любыми типами данных (например, int, double, пользовательские классы).
* Vector() {} – Создает пустой вектор.
* Vector(int n): vect(n, T()) – Создает вектор размера n, инициализируя все элементы значением по умолчанию для типа T (например, 0 для чисел).
* T& operator[](int index) { return vect[index]; } – Позволяет получать доступ к элементам вектора по индексу (без проверки границ).
* int Size() { return vect.size();} – Возвращает текущее количество элементов в векторе.
* void input() {for (auto& el : vect) std::cin >> el;} – Заполняет вектор значениями, введенными пользователем с клавиатуры.
* friend std::ostream& operator<<(std::ostream& os, Vector<T>& v) {for (auto& el : v.vect) os << el << "\t"; return os;} – Обеспечивает удобный вывод элементов вектора в поток (например, на экран) через std::cout.

**Класс Matrix:**

* Matrix(int n, int m) {this->vect.resize(n, Vector<T>(m));} – Создает матрицу размером n×m. Инициализирует все элементы значением по умолчанию для типа T
* int chislo\_strok() { return this->vect.size(); }

int chislo\_stolbcov() { return this->vect[0].Size(); } – Возвращают количество строк и столбцов соответственно.

* void Swap(int stroka1, int stroka2) {std::swap(this->vect[stroka1], this->vect[stroka2]);} – Меняет местами две указанные строки.
* void input() {for (auto& str : this->vect) str.input();} – Обеспечивает поэлементный ввод значений матрицы.

**Класс SLAU:**

* Right\_H - вектор правых частей системы
* reshenie - вектор для хранения решения
* solved - флаг, указывающий на наличие решения
* SLAU(Matrix<T>& A, Vector<T>& b): Matrix<T>(A), Right\_H(b), reshenie(b.Size()) – Принимает матрицу коэффициентов и вектор правых частей. Проверяет соответствие размеров.
* Resh() – Реализует прямой и обратный ход метода Гаусса. Включает поиск ведущего элемента для устойчивости. Выполняет проверку на вырожденность системы.
* Vector<T> poluchit\_reshenie() – Возвращает решение системы. При необходимости вызывает метод Resh().

**Main:**  
Блок try-catch – реализует механизм обработки исключений, который: защищает программу от аварийного завершения; предоставляет информативные сообщения об ошибках; позволяет корректно обрабатывать исключительные ситуации.

Блок try:

* + Содержит потенциально опасный код
  + Создание объекта SLAU и вызов метода решения
  + Вывод результатов

Блок catch:

* + Перехватывает исключения типа std::exception
  + Выводит сообщение об ошибке в стандартный поток ошибок (cerr)
  + Использует метод what() для получения текста ошибки

# Результаты экспериментов

Для проверки корректности работы программы были рассмотрены три примера систем линейных алгебраических уравнений (СЛАУ) с самостоятельным расчетом и программным решением. Во всех случаях результаты совпали с ожидаемыми значениями.

**Пример 1. Cистема 2×2**

Исходная система:

3x + 2y = 8

4x – y = 6

Прямой ход:

1-ую строку делим на 3: (1, 2/3, | 8/3)

От 2 строки отнимаем 1 строку, умноженную на 4: (0, -5/3 | -14/3)

2-ую строку делим на -5/3: (0, 1 | 2.8)

Обратный ход:

от 1 строки отнимаем 2 строку, умноженную на 2/3: x1=0.8, x2=2.8

Решение программы (рис.6):

Изображение выглядит как текст, Шрифт, снимок экрана

Автоматически созданное описание

Рис. 6 Решение программы.

Проверка:

3·0.8 + 2·2.8 = 2.4 + 5.6 = 8

4·0.8 + 2.8 = 3.2 + 2.8 = 6

Проверка выполнена успешно.

**Пример 2. Система 3×3**

Исходная система:

x + y + z = 6

2x + 3y + z = 14

x + 2y + 3z = 14

Прямой ход:

1-ую строку оставляем как есть: (1, 1, 1 | 6)

От 2 строки отнимаем 1 строку, умноженную на 2: (0, 1, -1 | 2)

От 3 строки отнимаем 1 строку: (0, 1, 2 | 8)

От 3 строки отнимаем 2 строку: (0, 0, 3 | 6)

3-ю строку делим на 3: (0, 0, 1 | 2)

Обратный ход:

из 2 строки: y - z = 2 → y = 4

из 1 строки: x + y + z = 6 → x = 0

x1=0, x2=4, x3=2

Решение программы (рис.7):

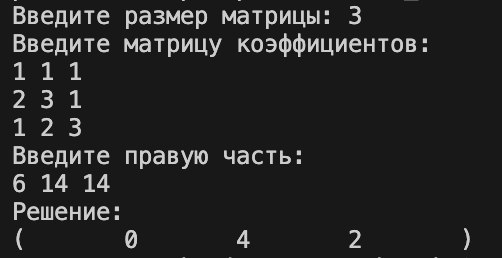


Рис. 7 Решение программы.

Проверка:

1·0 + 1·4 + 1·2 = 6

2·0 + 3·4 + 1·2 = 14

1·0 + 2·4 + 3·2 = 14

Проверка выполнена успешно.

**Пример 3. Вырожденная система**

Исходная система:

x + y = 2

2x + 2y = 4

Самостоятельный анализ:

Определитель матрицы равен 0 → система вырождена.

Второе уравнение является кратным первому → бесконечное число решений.

Решение программы (рис.8):

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рис. 8 Решение программы.

Решение программы совпало.

# Заключение

В ходе лабораторной работы была разработана программа для решения систем линейных уравнений методом Гаусса с выбором ведущего элемента. Реализованные классы Vector, Matrix и SLAU продемонстрировали эффективность при работе с матрицами различных размеров, обеспечивая точные вычисления благодаря устойчивому алгоритму и обработке особых случаев. Программа обладает удобным интерфейсом ввода-вывода и надежной системой обработки ошибок.

Работа имеет практическую ценность для вычислительной математики, предоставляя готовое решение фундаментальной задачи. Разработанный код может служить основой для более сложных алгоритмов или в дальнейшем быть интегрирован в другие проекты.

# Литература

1. Керниган Б., Ритчи Д., Фьюэр А. Язык программирования СИ //М.: Финансы и статистика. – 1992.
2. Роберт Лафоре, «Объектно-ориентированное программирование в C++».
3. Дж. Клейнберг, Э. Тардос. Алгоритмы: разработка и применение.
4. Р. Мартин. Чистый код.

# Приложение

Изображение выглядит как текст, снимок экрана, программное обеспечение, Мультимедийное программное обеспечение

Автоматически созданное описание

*Рис.9* **«**Lab\_№3.cpp»

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

*Рис. 10* **«**Lab\_№3.cpp»

Изображение выглядит как текст, снимок экрана, программное обеспечение

Автоматически созданное описание

*Рис.11* **«**Lab\_№3.cpp»