МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
**АДЫГЕЙСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ**

Инженерно-физический факультет
Кафедра автоматизированных систем обработки информации и
управления

Отчет по практике

Вариант 6
Нахождение ранга матрицы

1 курс, группа 1ИВТ АСОИУ

Выполнил:
_____ Э. А. Туов
«___» _____ 2024 г.

Руководитель:
_____ С. В. Теплоухов
«___» _____ 2024 г.

Майкоп, 2024 г.

Введение

В данной работе реализован алгоритм нахождения ранга матрицы с использованием метода Гаусса. Программа написана на языке C++ и использует динамические массивы для хранения матрицы.

Описание кода

Программа состоит из следующих частей:

1) Подключение необходимых библиотек.

2) Функция для вывода матрицы.

3) Функция для приведения матрицы к ступенчатому виду методом Гаусса.

4) Основная функция, в которой осуществляется ввод матрицы, вызов функций и вывод результатов.

Подключение библиотек

Подключаются стандартные библиотеки для ввода-вывода, обработки исключений и манипуляции выводом:

```
#include <iostream>
#include <iomanip>
#include <stdexcept>
using namespace std;
```

Функция для вывода матрицы

Функция `printMatrix` выводит матрицу в консоль, красиво форматируя вывод:

```
void printMatrix(double** matrix, int rows, int cols) {
    for (int i = 0; i < rows; ++i) {
        for (int j = 0; j < cols; ++j) {
            cout << setw(10) << matrix[i][j] << " ";
        }
        cout << endl;
    }
}
```

Функция приведения к ступенчатому виду

Функция `gaussianElimination` реализует метод Гаусса и возвращает ранг матрицы:

```
int gaussianElimination(double** matrix, int rows, int cols) {
    int rank = 0;
    for (int col = 0; col < cols; ++col) {
        int pivot = rank;
        while (pivot < rows && matrix[pivot][col] == 0) {
            ++pivot;
        }
        if (pivot == rows) {
            continue;
        }
        if (pivot != rank) {
            swap(matrix[rank], matrix[pivot]);
        }
        double pivotValue = matrix[rank][col];
        if (pivotValue == 0) {
            throw runtime_error("Pivot element is zero during normalization.");
        }
        for (int j = col; j < cols; ++j) {
            matrix[rank][j] /= pivotValue;
        }
        for (int i = rank + 1; i < rows; ++i) {
            double factor = matrix[i][col];
            for (int j = col; j < cols; ++j) {
                matrix[i][j] -= factor * matrix[rank][j];
            }
        }
        ++rank;
    }
    return rank;
}
```

Основная функция

Основная функция организует ввод данных, вызов функций и вывод результатов:

```
int main() {
```

```
    try {
        int rows, cols;
        cout << "Введите количество строк и столбцов матрицы: ";
        cin >> rows >> cols;
        if (rows <= 0 || cols <= 0) {
            throw invalid_argument("Количество строк и столбцов должно быть положит
        }
        double** matrix = new double*[rows];
        for (int i = 0; i < rows; ++i) {
            matrix[i] = new double[cols];
        }
        cout << "Введите элементы матрицы:\n";
        for (int i = 0; i < rows; ++i) {
            for (int j = 0; j < cols; ++j) {
                if (!(cin >> matrix[i][j])) {
                    throw runtime_error("Ошибка ввода. Пожалуйста, вводите только ч
                }
            }
        }
        cout << "Исходная матрица:\n";
        printMatrix(matrix, rows, cols);
        int rank = gaussianElimination(matrix, rows, cols);
        cout << "Матрица после приведения к ступенчатому виду:\n";
        printMatrix(matrix, rows, cols);
        cout << "Ранг матрицы: " << rank << endl;
        for (int i = 0; i < rows; ++i) {
            delete[] matrix[i];
        }
        delete[] matrix;
    } catch (const exception& e) {
        cerr << "Ошибка: " << e.what() << endl;
        return 1;
    }
    return 0;
}
```