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

Программная реализация численного метода
Текст из задания по варианту

1 курс, группа 1ИВТ АСОИУ

Выполнил:
_____В. А. Сапунов
«06» 06 2024 г.

Руководитель:
_____С. В. Теплоухов
«06» 06 2024 г.

Майкоп, 2024 г.

1. Введение

1) Текстовая формулировка задачи
2) Пример кода, решающего данную задачу
3) График
4) Скриншот программы

Пример приведен в пункте 2 на стр. 1.

2. Ход работы

2.1. Код приложения

```
#include <iostream>
#include <vector>
#include <iomanip>
#include <limits>

using namespace std;

void printMatrix(const vector<vector<double>>& matrix) {
    int rows = matrix.size();
    int cols = matrix[0].size();

    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            cout << setw(8) << matrix[i][j] << " ";
        }
        cout << endl;
    }
}
vector<vector<double>> inverseMatrix(vector<vector<double>>& matrix){
int n = matrix.size();

    vector<vector<double>> extendedMatrix(n, vector<double>(2 * n));
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            extendedMatrix[i][j] = matrix[i][j];
        }
        extendedMatrix[i][i + n] = 1;
    }
```

```
for (int i = 0; i < n; i++) {
    int pivotRow = i;
    while (pivotRow < n && abs(extendedMatrix[pivotRow][i])< 1e-6) {
        pivotRow++;
    }

    if (pivotRow == n) {
        continue;
    }

    if (pivotRow != i) {
        swap(extendedMatrix[i], extendedMatrix[pivotRow]);
    }

    double pivot = extendedMatrix[i][i];
    for (int j = i; j < 2 * n; j++) {
        extendedMatrix[i][j] /= pivot;
    }

    for (int k = i + 1; k < n; k++) {
        double factor = extendedMatrix[k][i];
        for (int j = i; j < 2 * n; j++) {
            extendedMatrix[k][j] -= factor * extendedMatrix[i][j];
        }
    }
}

for (int i = 0; i < n; i++) {
    bool allZero = true;
    for (int j = 0; j < n; j++) {
        if (abs(extendedMatrix[i][j]) > 1e-6) {
            allZero = false;
            break;
        }
    }
    if (allZero) {
        cout <<"Матрица вырождена! Обратная матрица не существует." << endl;
        return {};
    }
}
```

```
    for (int i = n - 1; i >= 0; i--) {
        for (int k = i - 1; k >= 0; k--) {
            double factor = extendedMatrix[k][i];
            for (int j = i; j < 2 * n; j++) {
                extendedMatrix[k][j] -= factor * extendedMatrix[i][j];
            }
        }
    }

    vector<vector<double>> inverse(n, vector<double>(n));
    for (int i = 0; i < n; i++) {
        for (int j = n; j < 2 * n; j++) {
            inverse[i][j - n] = extendedMatrix[i][j];
        }
    }

    return inverse;
}

int main() {
    setlocale(0, "ru");
    int choice;
    int n;
    vector<vector<double>> matrix;

    do {
        cout << "\nМеню:" << endl;
        cout << "1. Ввести новую матрицу" << endl;
        cout << "2. Найти обратную матрицу" << endl;
        cout << "3. Выход" << endl;
        cout << "Введите ваш выбор: ";
        cin >> choice;
        vector<vector<double>> inverse = inverseMatrix(matrix);
        switch (choice) {
                case 1:
            cout << "Введите размерность матрицы: ";
            cin >> n;

            if (n <= 0) {
cout <<"Некорректная размерность матрицы! Введите положительное число." << endl;
                break;
```

```
            }

            matrix.resize(n, vector<double>(n));
            cout << "Введите элементы матрицы:" << endl;
            for (int i = 0; i < n; i++) {
                for (int j = 0; j < n; j++) {
                    if (!(cin >> matrix[i][j])) {
                        cout << "Ошибка ввода данных! Введите числовые значения." << endl;
                        cin.clear();
                        cin.ignore(numeric_limits<streamsize>::max(), '\n');
                        break;
                    }
                }
            }
            cout << "Исходная матрица:" << endl;
            printMatrix(matrix);
            break;

        case 2:
            if (matrix.empty()) {
                cout << "Сначала введите матрицу." << endl;
                break;
            }
            if (inverse.empty()) {

матрица не существует!" << endl;
            }
            else {
                cout << "Обратная матрица:" << endl;
                printMatrix(inverse);
            }
            break;

        case 3:
            cout << "Выход из программы." << endl;
            break;

        default:
            cout << "Неверный выбор!" << endl;
        }
    } while (choice != 3);
```

```
return 0;
}
```

2.2. Метод вычисления обратной матрицы

Вычисление обратной матрицы ([[1. -2. 1.] [2. 1. -1.] [3. 2. -2.]]):
Алгоритм нахождения обратной матрицы методом исключения неизвестных Гаусса:

1. К матрице А приписать единичную матрицу того же порядка.

2. Полученную сдвоенную матрицу преобразовать так, чтобы в левой её части получилась единичная матрица, тогда в правой части на месте единичной матрицы автоматически получится обратная матрица. Матрица А в левой части преобразуется в единичную матрицу путём элементарных преобразований матрицы.

3. Если в процессе преобразования матрицы А в единичную матрицу в какой-либо строке или в каком-либо столбце окажутся только нули, то определитель матрицы равен нулю, и, следовательно, матрица А будет вырожденной, и она не имеет обратной матрицы. В этом случае дальнейшее нахождение обратной матрицы прекращается.

3. Изображение с примером вычисления обратной матрицы и результат выполнения программы

Пример вычисления обратной матрицы на рис. 1.
Результат выполнения программы на рис. 2.

1. $|A| = \begin{pmatrix} 1 & -2 & 1 \\ 2 & 1 & -1 \\ 3 & 2 & -2 \end{pmatrix}$
2. $|A| \neq 0$ – матрица невырожденная
3. $A_{11} = (-1)^{1+1} \times \begin{vmatrix} 1 & -1 \\ 2 & -2 \end{vmatrix} = 0$; $A_{21} = (-1)^{2+1} \times \begin{vmatrix} -2 & 1 \\ 2 & -2 \end{vmatrix} = -2$; $A_{31} = (-1)^{3+1} \times \begin{vmatrix} -2 & 1 \\ 1 & -1 \end{vmatrix} = 1$;
   $A_{12} = (-1)^{1+2} \times \begin{vmatrix} 2 & -1 \\ 3 & -2 \end{vmatrix} = 1$; $A_{22} = (-1)^{2+2} \times \begin{vmatrix} 1 & 1 \\ 3 & -2 \end{vmatrix} = -5$; $A_{32} = (-1)^{3+2} \times \begin{vmatrix} 1 & 1 \\ 2 & -1 \end{vmatrix} = 3$;
   $A_{13} = (-1)^{1+3} \times \begin{vmatrix} 2 & 1 \\ 3 & 2 \end{vmatrix} = 1$; $A_{23} = (-1)^{2+3} \times \begin{vmatrix} 1 & -2 \\ 3 & 2 \end{vmatrix} = -8$; $A_{33} = (-1)^{3+3} \times \begin{vmatrix} 1 & -2 \\ 2 & 1 \end{vmatrix} = 5$;
4. $A^{-1} = \frac{1}{-1} \times \begin{pmatrix} 0 & -2 & 1 \\ 1 & -5 & 3 \\ 1 & -8 & 5 \end{pmatrix} = \begin{pmatrix} 0 & 2 & -1 \\ -1 & 5 & -3 \\ -1 & 8 & -5 \end{pmatrix}$
5. Проверка: $A^{-1} = \begin{pmatrix} 1 & -2 & 1 \\ 2 & 1 & -1 \\ 3 & 2 & -2 \end{pmatrix} \times \begin{pmatrix} 0 & 2 & -1 \\ -1 & 5 & -3 \\ -1 & 8 & -5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Рисунок 1

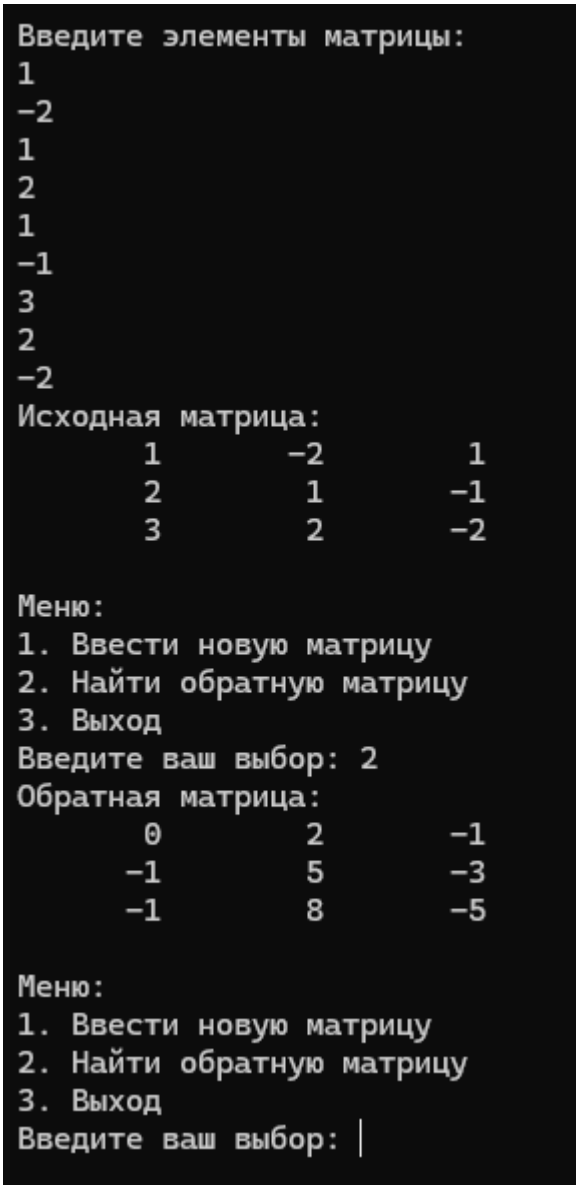
Введите элементы матрицы:
1
-2
1
2
1
-1
3
2
-2
Исходная матрица:
1 -2 1
2 1 -1
3 2 -2
Меню:
1. Ввести новую матрицу
2. Найти обратную матрицу
3. Выход
Введите ваш выбор: 2
Обратная матрица:
0 2 -1
-1 5 -3
-1 8 -5
Меню:
1. Ввести новую матрицу
2. Найти обратную матрицу
3. Выход
Введите ваш выбор:

Рисунок 2

4. Пример библиографических ссылок

Для изучения «внутренностейk TEX необходимо изучить [1], а для использования LATEX лучше почитать [2, 3].

Список литературы:

[1] Кнут Д.Э. Всё про TEX. — Москва: Изд. Вильямс, 2003 г. 550 с.
[2] Львовский С.М. Набор и верстка в системе LATEX. — 3-е издание, исправленное и дополненное, 2003 г.
[3] Воронцов К.В. LATEX в примерах. 2005 г.

Рис. 1. Парабола

Список литературы

[1] Кнут Д.Э. Всё про TEX. — Москва: Изд. Вильямс, 2003 г. 550 с.

[2] Львовский С.М. Набор и верстка в системе LATEX. — 3-е издание, исправленное и дополненное, 2003 г.

[3] Воронцов К.В. LATEX в примерах. 2005 г.