

Лабораторная работа № 9

1. Тема: численные методы решения систем линейных алгебраических уравнений. Метод Гаусса.
2. Постановка задачи:
решить систему линейных алгебраических уравнений, представленную матрицей:

$$\left(\begin{array}{cccc|c} 5 & 7 & 6 & 5 & 23 \\ 7 & 10 & 8 & 7 & 32 \\ 6 & 8 & 10 & 9 & 33 \\ 5 & 7 & 9 & 10 & 31 \end{array} \right)$$

3. Мат. модель:

Этап прямого хода:

$$\widetilde{a}_{ij} = \frac{a_{ij}}{a_{ii}}, i = 1 \dots (n-1), j = i \dots (n+1)$$

$$\widetilde{a}_{kj} = a_{kj} - \widetilde{a}_{ij}a_{ki}, k = (i+1) \dots n$$

Этап обратного хода:

$$x_n = \frac{\widetilde{a}_{n(n+1)}}{\widetilde{a}_{nn}}$$

$$x_i = a_{i(n+1)} - \sum_{j=i+1}^n a_{ij}x_j, i = (n-1) \dots 1$$

4. Список идентификаторов: (в скобках указаны функции, в которых находится переменная)

Имя	Тип	Смысл
M	double	Указатель на массив в функциях
n	int	Количество строк матрицы
m	int	Количество столбцов матрицы, включая столбец свободных членов
i, j, k	int	Переменные для массивов
Aki	double	Новый элемент строки после преобразования
Aii	double	Элемент строки после деления
A	double	Двумерный массив
X	double	Одномерный массив результатов

5. Код программы:

```
#include <iostream>
#include <cstdio>
#include <cstdlib>
```

```
#include <iomanip>
```

```
using namespace std;
```

```
void outputMatrix1(double* M, int n){  
    for (int i = 0; i < n; i++){  
        cout << setw(6) << left << M[i] << " ";  
    }  
}
```

```
void outputMatrix2(double** M, int n, int m){  
    for (int i = 0; i < n; i++){  
        for (int j = 0; j < m - 1; j++){  
            cout << setw(6) << left << M[i][j] << " ";  
        }  
        cout << "|" << setw(6) << right << M[i][m - 1] << endl;  
    }  
}
```

```
void createMatrix(double** M, int n, int m){  
    for (int i = 0; i < n; i++){  
        M[i] = new double[m];  
        for (int j = 0; j < m; j++){  
            M[i][j] = 0;  
        }  
    }  
}
```

```
void inputMatrix(double** M, int n){  
    for (int i = 0; i < n; i++){  
        for (int j = 0; j < n; j++){  
            cin >> M[i][j];  
        }  
    }  
}
```

```
void inputMatrixColumn(double** M, int n, int m){  
    for (int i = 0; i < n; i++){  
        cin >> M[i][m - 1];  
    }  
}
```

```

void changeMatrix(double** M, int k, int m){
    double q;
    for (int j = 0; j < m; j++){
        q = M[k][j];
        M[k][j] = M[k + 1][j];
        M[k + 1][j] = q;
    }
}

int main(){
    int n, m, i, j, k;
    double Aki, Aii;
    cout << "Input number of rows and columns: "; cin >> n;
    m = n + 1;
    double** A = new double*[n]; createMatrix(A, n, m); cout << endl;
    cout << "Input matrix: \n"; inputMatrix(A, n); cout << endl;
    cout << "Input matrix's column of free members: \n";
    inputMatrixColumn(A, n, m);
    cout << endl;
    cout << "Matrix: " << endl;
    outputMatrix2(A, n, m);
    cout << endl;
    for (i = 0; i < n - 1; i++){
        if (A[i][i] == 0){
            changeMatrix(A, i, m);
        }
    }
    for (i = 0; i < n; i++){
        Aii = A[i][i];
        for (j = 0; j < m; j++){
            A[i][j] /= Aii;
        }
        for (k = i + 1; k < n; k++){
            Aki = A[k][i];
            for (j = i; j < m; j++){
                A[k][j] -= Aki*A[i][j];
            }
        }
    }
    cout << "Middle matrix: " << endl;
    outputMatrix2(A, n, m);
    cout << endl;
}

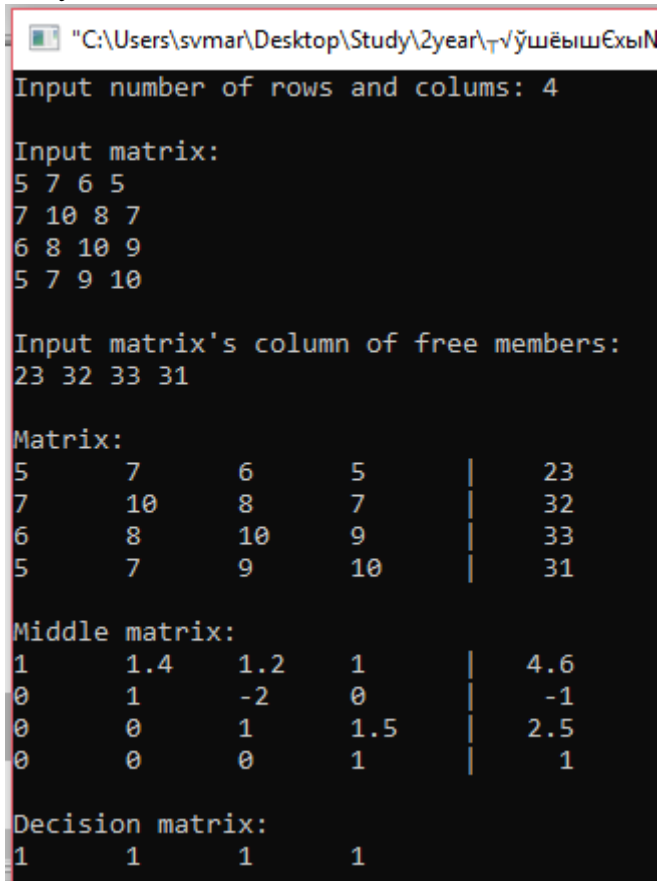
```

```

double X[n] = {0};
X[n - 1] = A[n - 1][m - 1]/A[n - 1][n - 1];
for (i = n - 2; i >= 0; i--){
    X[i] = A[i][m - 1];
    for(j = i + 1; j <= m - 2; j++){
        X[i] -= (A[i][j]*X[j]);
    }
    X[i] /= A[i][i];
}
cout << "Decision matrix: " << endl;
outputMatrix1(X, n);
cout << endl;
return 0;
}

```

6. Результаты:



The screenshot shows the output of a C++ program. It starts with a prompt to input the number of rows and columns, which is 4. Then, it prompts for the input matrix, which is a 4x4 matrix. Next, it prompts for the column of free members, which is a 4x1 vector. The program then displays the augmented matrix (Matrix:) with a vertical line separating the coefficients from the free members. It then displays the middle matrix (Middle matrix:) which is the result of row reduction. Finally, it displays the decision matrix (Decision matrix:), which is a 4x4 identity matrix.

```

"C:\Users\svmar\Desktop\Study\2year\7\7\ШЕЫШЕХЫN
Input number of rows and columns: 4

Input matrix:
5 7 6 5
7 10 8 7
6 8 10 9
5 7 9 10

Input matrix's column of free members:
23 32 33 31

Matrix:
5      7      6      5      |      23
7      10     8      7      |      32
6      8      10     9      |      33
5      7      9      10     |      31

Middle matrix:
1      1.4    1.2    1      |      4.6
0      1      -2     0      |      -1
0      0      1      1.5    |      2.5
0      0      0      1      |      1

Decision matrix:
1      1      1      1

```