**Министерство образования и науки Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего профессионального образования  
«Московский государственный технический университет имени Н.Э. Баумана»  
(МГТУ им. Н.Э. Баумана)**

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | **«Информатики и систем управления»** |
| **КАФЕДРА** | **5-13** |

**ОТЧЕТ**

***ПО ЛАБОРАТОРНОЙ РАБОТЕ № 10***

ПО ДИСЦИПЛИНЕ:«Основы программирования»

Студент Гаврилов Л.Я.

(Группа) (Подпись, дата) (И.О. Фамилия)

Проверил Козлов А.Д.

(Подпись, дата) (И.О. Фамилия)

*2018  г.*

# Задания на лабораторную работу

## Задание 1

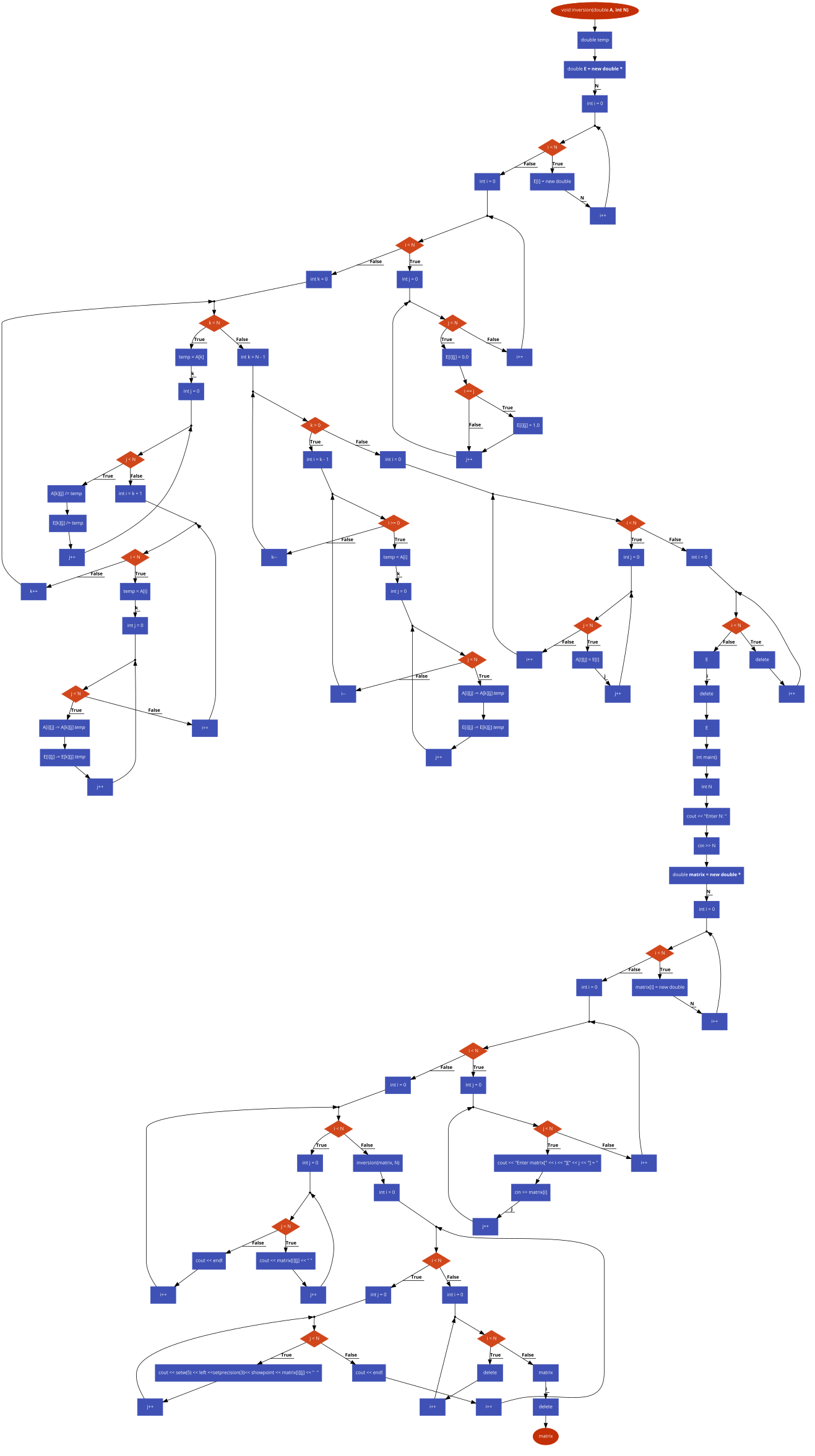
Вычисление обратной матрицы методом Гаусса-Жордана

Создать функцию для вычисления обратной матрицы по методу Гаусса-Жордана. Размер матрицы передавать в функцию в качестве параметра. Для упрощения алгоритма следует присоединить единичную матрицу справа к исходной и выполнять все преобразования над объединенной матрицей размером N\*2N. Обратная матрица получится на месте единичной в столбцах N…2N, а на месте исходной матрицы в столбцах 0…(N-1) должна получиться единичная матрица.

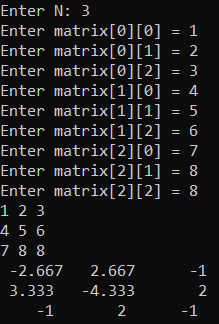
# Описание разработанных программ

## Задание 1

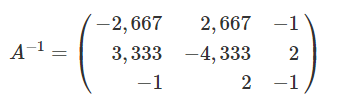
### Описание используемых алгоритмов

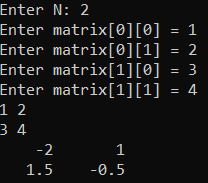


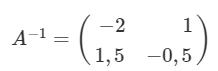
### Пример выполнения программы

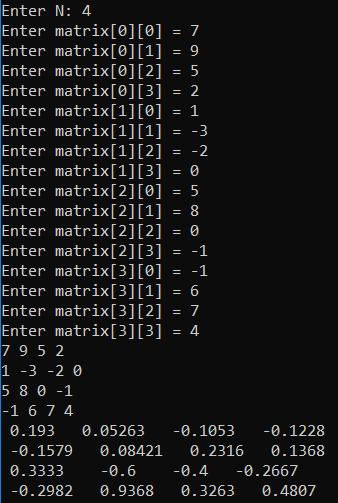


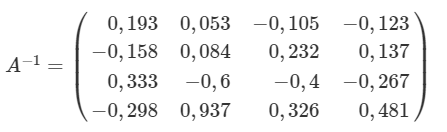
Доказательство данного примера и последующих:











### Текст программы

#include "pch.h"

#include <iostream>

#include <iomanip>

using namespace std;

void inversion(double \*\*A, int N)

{

double temp;

double \*\*E = new double \*[N];

for (int i = 0; i < N; i++)

E[i] = new double[N];

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

{

E[i][j] = 0.0;

if (i == j)

E[i][j] = 1.0;

}

for (int k = 0; k < N; k++)

{

temp = A[k][k];

for (int j = 0; j < N; j++)

{

A[k][j] /= temp;

E[k][j] /= temp;

}

for (int i = k + 1; i < N; i++)

{

temp = A[i][k];

for (int j = 0; j < N; j++)

{

A[i][j] -= A[k][j] \* temp;

E[i][j] -= E[k][j] \* temp;

}

}

}

for (int k = N - 1; k > 0; k--)

{

for (int i = k - 1; i >= 0; i--)

{

temp = A[i][k];

for (int j = 0; j < N; j++)

{

A[i][j] -= A[k][j] \* temp;

E[i][j] -= E[k][j] \* temp;

}

}

}

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

A[i][j] = E[i][j];

for (int i = 0; i < N; i++)

delete[] E[i];

delete[] E;

}

int main()

{

int N;

cout << "Enter N: ";

cin >> N;

double \*\*matrix = new double \*[N];

for (int i = 0; i < N; i++)

matrix[i] = new double[N];

for (int i = 0; i < N; i++)

for (int j = 0; j < N; j++)

{

cout << "Enter matrix[" << i << "][" << j << "] = ";

cin >> matrix[i][j];

}

for (int i = 0; i < N; i++) {

for (int j = 0; j < N; j++) {

cout << matrix[i][j] << " ";

}

cout << endl;

}

inversion(matrix, N);

for (int i = 0; i < N; i++)

{

for (int j = 0; j < N; j++)

cout << " " << setw(5) << right << setprecision(4) <<matrix[i][j] << " ";

cout << endl;

}

for (int i = 0; i < N; i++)

delete[] matrix[i];

delete[] matrix;

system("pause");

return 0;

}

# Выводы

**Повторение всего изученного материала.**