**Лабораторная работа №3**

***Условие:***

1. Проверить, есть ли среди элементов главной диагонали матрицы *A* (5×5) отрицательные элементы, а среди элементов главной диаго­нали матрицы *B* (3×3) — элементы, меньшие 2,7.
2. Если максимальный элемент матрицы, находящийся на главной диагонали больше 10, то отсортировать по возрастанию элементы второй строки, иначе – третьего столбца. Сортировка методом обмена.

Выбор назначения подпрограмм:

Для упрощения работы оптимальным будет создание одной подпрограммы выполняющей с помощью функций условия 1 и 2 задачи.

Алгоритм ввода матриц одинаков для массивов А и В, поэтому оформим ее в функцию с названием matrix\_create и применить к матрицам А и В. Условие первой задачи оформим в функцию matrix\_first. Условие второй задачи оформим в функцию matrix\_second. Она одинакова для массивов A и B, поэтому ее создание упрощает понимание кода и сокращает количество ошибок.

***Шаблоны выбранных функций:***

void matrix\_create(int n, double A[][5]) – Создает матрицу

void matrix\_first(int n, double A[][5], int m, double B[][5]) – Выполняет первое условие

void matrix\_second(int n, double A[][5]) – Выполняет второе условие

***Cостав данных функции main:***

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Смысл** | **Тип** | **Структура** |
| Исходные данные | | | |
| **A** | заданные матрицы | вещественный | двумерный массив **с числом столбцов и строк 5** |
| **B** |

***Состав данных функции matrix\_create:***

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Смысл** | **Тип** | **Структура** |
| Исходные данные | | | |
| **A** | заданная матрица | вещественный | двумерный массив **с числом столбцов и строк 5** |
| **n** | величина квадратной матрицы | целый | простая переменная |
| **create\_flag** | выбор ввода матрицы А | символьный | простая переменная |
| Промежуточные данные | | | |
| **i** | счетчик цикла | целый | простая переменная |
| **j** | счетчик цикла | целый | простая переменная |

***Состав данных функции matrix\_first:***

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Смысл** | **Тип** | **Структура** |
| Исходные данные | | | |
| **A** | заданная матрица | вещественный | двумерный массив **с числом столбцов и строк 5** |
| **n** | величина квадратной матрицы A | целый | простая переменная |
| **B** | заданная матрица | вещественный | двумерный массив **с числом столбцов и строк 5** |
| **m** | величина квадратной матрицы B | целый | простая переменная |
| Промежуточные данные | | | |
| **i** | счетчик цикла | целый | простая переменная |
| **j** | счетчик цикла | целый | простая переменная |
| **a** | счетчик отрицательных элементов матрицы А | целый | простая переменная |
| **b** | счетчик элементов матрицы В, меньших 2,7 | целый | простая переменная |

***Состав данных функции matrix\_second:***

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя** | **Смысл** | **Тип** | **Структура** |
| Исходные данные | | | |
| **A** | заданная матрица | вещественный | двумерный массив **с числом столбцов и строк 5** |
| **n** | величина квадратной матрицы | целый | простая переменная |
| Промежуточные данные | | | |
| **i** | счетчик цикла | целый | простая переменная |
| **j** | счетчик цикла | целый | простая переменная |
| **max** | максимум диагонали матрицы А | вещественный | простая переменная |
| **temp** | переменная для обмена значений элементов массива | вещественный | простая переменная |

***форма ввода для функции matrix\_create:***

|  |
| --- |
| Создать матрицу автоматически? (Y-да/N-нет)  <Y/N>  Введите элементы массива А:  Строка № <i+1>  <A[i][j]> |

***форма вывода для функции matrix\_create:***

|  |
| --- |
| Полученный массив:  <A[i][j]> |

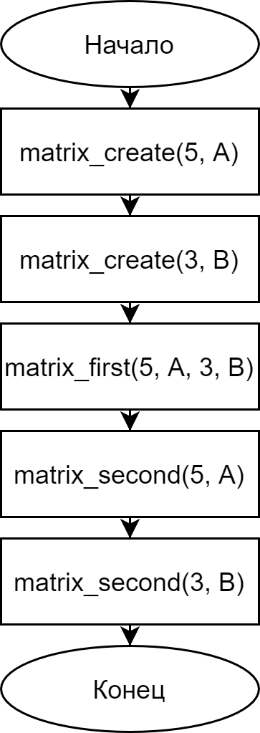
***форма вывода для функции matrix\_first:***

|  |
| --- |
| Количество отрицательных элементов диагонали матрицы А: <a>  Количество элементов диагонали матрицы В, меньших 2.7: <b> |

***форма вывода для функции matrix\_second:***

|  |
| --- |
| Максимальный элемент диагонали больше/меньше 10  Полученный массив:  <A[i][j]> |

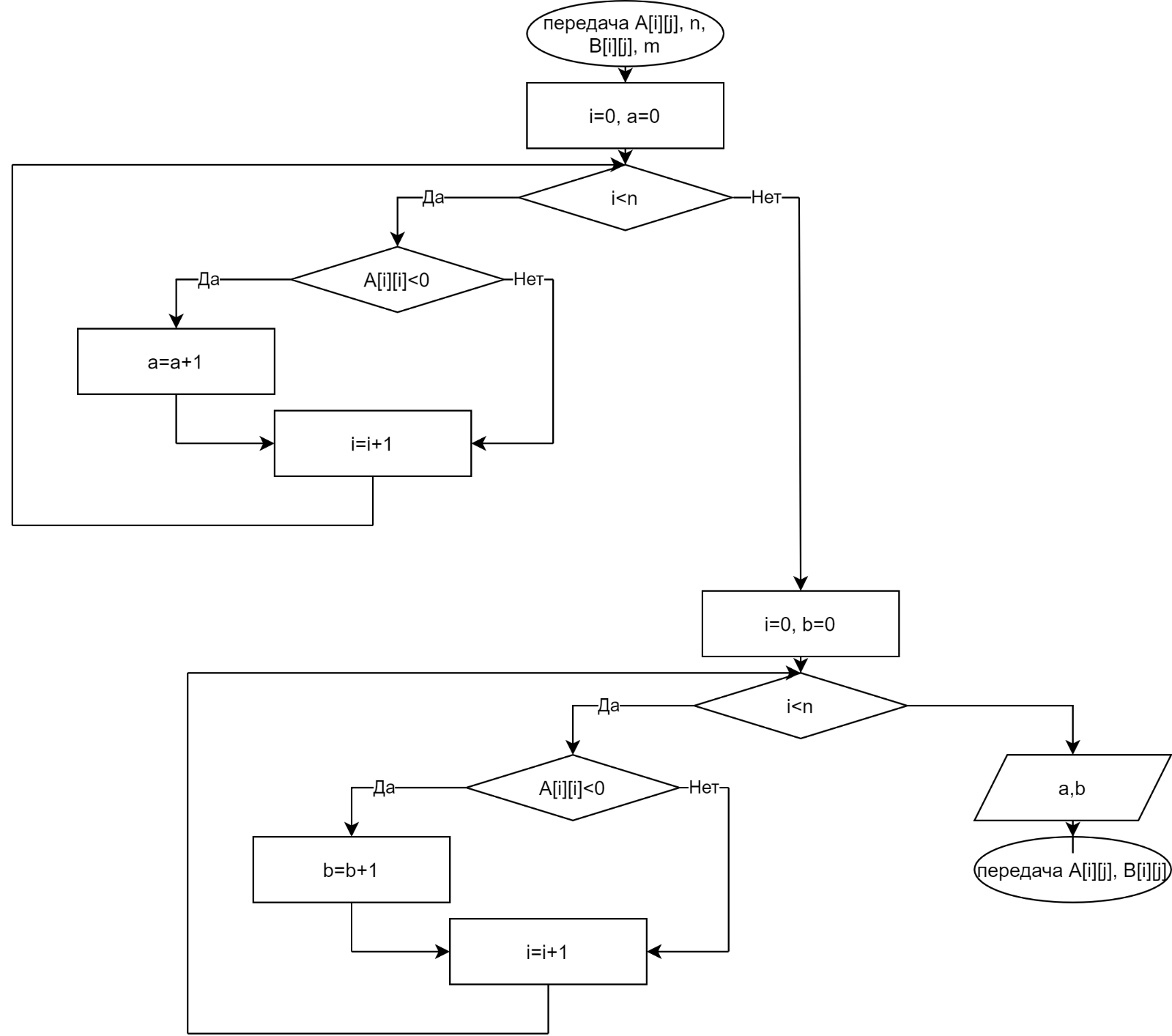
***Блок-схема алгоритма функции main:***



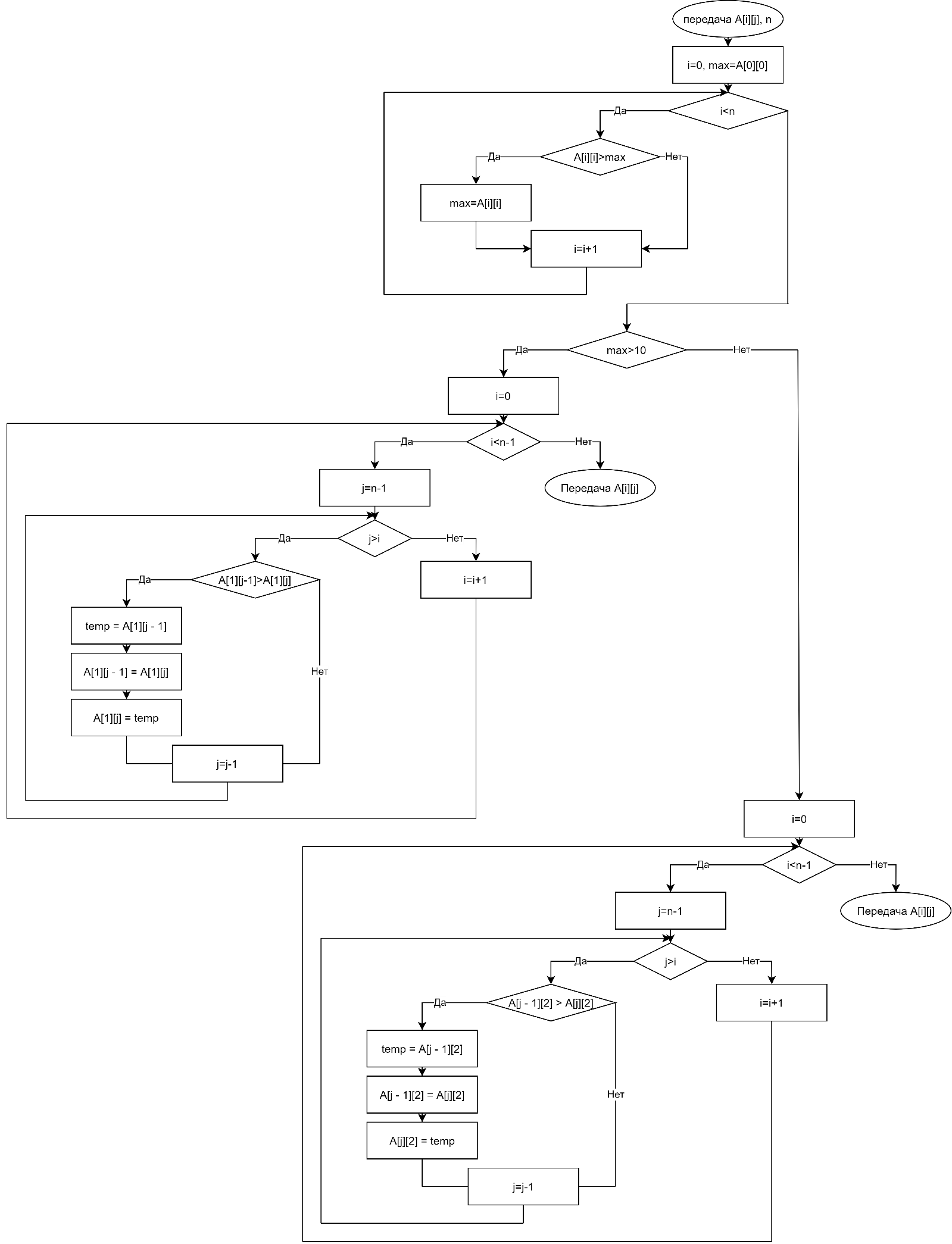
***Блок-схема алгоритма функции matrix\_create:***



***Блок-схема алгоритма функции matrix\_first:***



***Блок-схема алгоритма функции matrix\_second:***



***Код программы:***

#include <iostream>

#include <ctime>

using namespace std;

/\*

1. Проверить, есть ли среди элементов главной диагонали матрицы A (5×5) отрицательные элементы,

а среди элементов главной диаго¬нали матрицы B (3×3) — элементы, меньшие 2,7.

2. Если максимальный элемент матрицы, находящийся на главной диагонали больше 10,

то отсортировать по возрастанию элементы второй строки, иначе – третьего столбца. Сортировка методом обмена.

\*/

void matrix\_create(int n, double A[][5]);

void matrix\_first(int n, double A[][5], int m, double B[][5]);

void matrix\_second(int n, double A[][5]);

int main()

{

setlocale(LC\_ALL, "Russian");

srand(static\_cast<unsigned int>(time(0))); // Привязка к времени системных часов для получения "рандомных" чисел

double A[5][5], B[5][5];

cout << "матрица A:" << endl;

matrix\_create(5, A);

cout << endl << "матрица B:" << endl;

matrix\_create(3, B);

cout << endl;

matrix\_first(5, A, 3, B);

cout << endl << "матрица A:" << endl;

matrix\_second(5, A);

cout << endl;

cout << "матрица B:" << endl;

matrix\_second(3, B);

system("pause");

return 0;

}

void matrix\_create(int n, double A[][5])

{

cout << "Создать матрицу автоматически? (Y-да/N-нет)" << endl;

char create\_flag;

cin >> create\_flag;

try

{

if ((create\_flag != 'Y') && (create\_flag != 'N'))

throw "Необходимо выбрать Y-да/N-нет!";

// Автоматическое создание матриц A и В

if (create\_flag == 'Y') {

for (int i = 0; i < n; i += 1) {

for (int j = 0; j < n; j += 1) {

A[i][j] = (500 - rand() % 1000) / 10;

}

}

}

// Ручное создание матриц А и B

if (create\_flag == 'N') {

cout << "Введите элементы массива А:" << endl;

for (int i = 0; i < n; i += 1) {

cout << "Строка №" << i + 1 << ":" << endl;

for (int j = 0; j < n; j += 1) {

cin >> A[i][j];

}

}

}

// Вывод массивов

cout << "Полученный массив: " << endl;

for (int i = 0; i < n; i += 1) {

for (int j = 0; j < n; j += 1) {

cout << A[i][j] << "\t";

}

cout << endl;

}

}

catch (const char\* exception)

{

cerr << "Ошибка: " << exception << '\n';

}

}

void matrix\_first(int n, double A[][5], int m, double B[][5])

{

int a = 0, b = 0;

for (int i = 0; i < n; i += 1) {

if (A[i][i] < 0) a = a + 1;

}

for (int i = 0; i < m; i += 1) {

if (B[i][i] < 2.7) b = b + 1;

}

if (a != 0) cout << "Количество отрицательных элементов диагонали матрицы А: " << a << endl;

else cout << "Отрицательных элементов в диагонали матрицы А нет" << endl;

if (b != 0) cout << "Количество элементов диагонали матрицы В, меньших 2.7: " << b << endl;

else cout << "Элементов диагонали матрицы В, меньших 2.7 нет" << endl;

}

void matrix\_second(int n, double A[][5]) {

double max = A[0][0];

for (int i = 1; i < n; i += 1) {

if (A[i][i] > max) max = A[i][i];

}

if (max > 10) {

cout << "Максимальный элемент диагонали больше 10" << endl;

for (int i = 0; i < n - 1; i++)

{

for (int j = (n - 1); j > i; j--)

{

if (A[1][j - 1] > A[1][j])

{

double temp = A[1][j - 1];

A[1][j - 1] = A[1][j];

A[1][j] = temp;

}

}

}

}

else {

cout << "Максимальный элемент диагонали меньше 10" << endl;

for (int i = 0; i < n - 1; i++)

{

for (int j = (n - 1); j > i; j--)

{

if (A[j - 1][2] > A[j][2])

{

double temp = A[j - 1][2];

A[j - 1][2] = A[j][2];

A[j][2] = temp;

}

}

}

}

cout << "Полученный массив: " << endl;

for (int i = 0; i < n; i += 1) {

for (int j = 0; j < n; j += 1) {

cout << A[i][j] << "\t";

}

cout << endl;

}

}