Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

Лабораторная работа 14

По дисциплине «Основы алгоритмизации и программирования»

На тему «Многомерные массивы»

Выполнил:

Студент 1 курса 9 группы

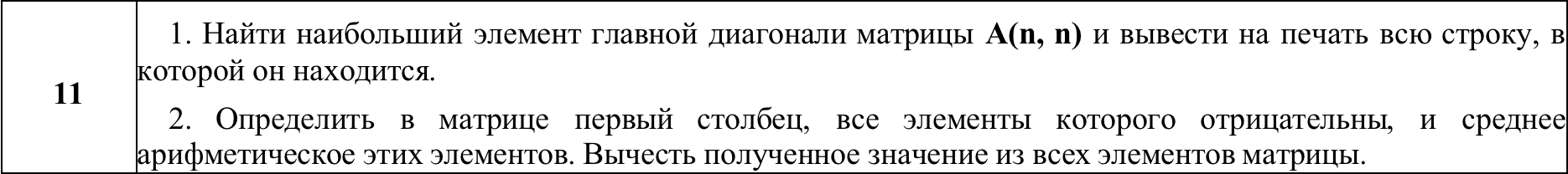
Павлович Ян Андреевич

Преподаватель: асс. Андронова М.В.

2023, Минск

Вариант 11

1. Выполнить задания из таблицы ниже, используя ***индексы*** для доступа к элементам массивов в первой программе и ***указатели*** − во второй программе.



1. Найти наибольший элемент главной диагонали матрицы **A(n, n)** и вывести на печать всю строку, в которой он находится.

#include <iostream> //для работы с вводом/выводом

**using** **namespace** std; //объявляем программе, что мы используем пространство имен std

**int** main() {

    setlocale(LC\_CTYPE, "Russian"); //возможность работы с русским языком

**int** max = 0, line = 0;

**int** A[3][3] = { {1, 2, 3}, {4, 5, 3}, {4, 2, 1} };

    //с помощью цикла for ищем на главной диагонали наибольший элемент

**for** (**int** i = 1; i < 4; i++) {

**if** (A[i][i] > max) {

            max = A[1][1];

            line++;

        }

    }

    cout << "Наибольший элемент главной диагонали матрицы: " << max << endl;

    //с помощью цикла for выводим строку с наибольшим элементом

    cout << "Строка с наибольшим элементом: ";

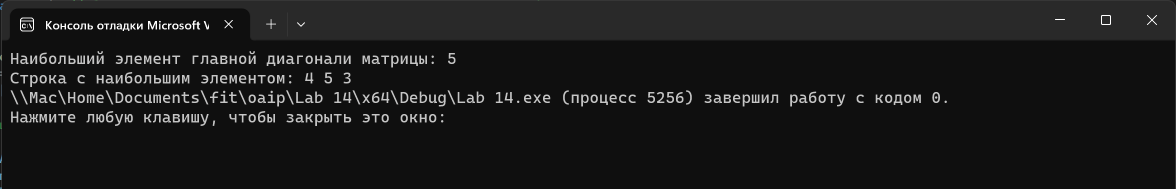
**for** (**int** i = 0; i < 3; i++) {

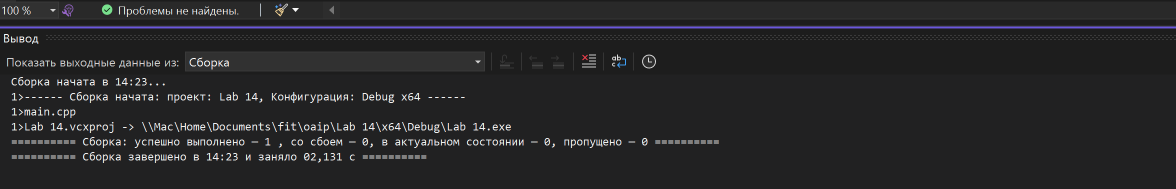
        cout << A[line][i] << " ";

    }

**return** 0;

}





1. Определить в матрице первый столбец, все элементы которого отрицательны, и среднее арифметическое этих элементов. Вычесть полученное значение из всех элементов матрицы.

#include <iostream> //для работы с вводом/выводом

**using** **namespace** std; //объявляем программе, что мы используем пространство имен std

**int** main() {

    setlocale(LC\_CTYPE, "Russian"); //возможность работы с русским языком

**int** sum = 0, col = 0, sr = 0;

**int** A[3][3] = {

        {-1, -2, 3},

        {4, -5, -3},

        {-4, -2, 1} };

    //с помощью цикла for ищем все отрицательные элементы в столбце

**for** (**int** i = 0; i < 3; i++) {

**for** (**int** j = 0; j < 3; j++) {

**if** ( \*(A[j] + i) >= 0) {

                sum = 0;

**break**;

            }

**else** {

                sum++;

            }

        }

**if** (sum == 3) {

            col = i;

**break**;

        }

    }

    //находим среднее арифмитическое

**for** (**int** i = 0; i < 3; i++) {

        sr+= \*(A[i] + col);

    }

    sr = sr / 3;

    cout << "Среднее арифметическое этих элементов: " << sr << endl;

    //с помощью цикла for выводим все элементы, отняв среднее арифмитическое

    cout << "При вычитании из всех элементов среднее арифметическое: ";

**for** (**int** i = 0; i < 3; i++) {

**for** (**int** j = 0; j < 3; j++) {

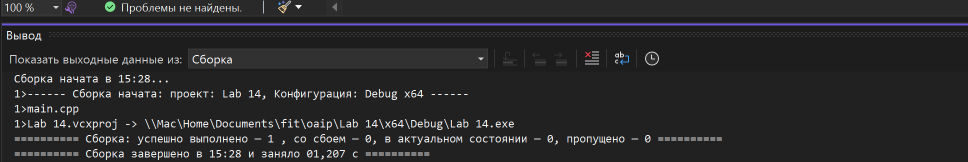
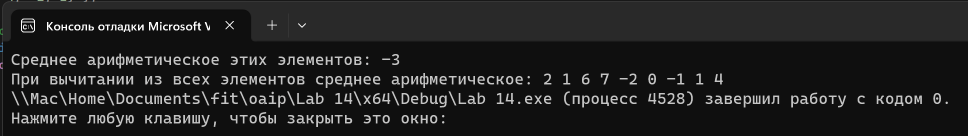
            cout << \*(A[i] + j) - sr << " ";

        }

    }

**return** 0;

}



**Дополнительные задания**

1. Дана квадратная матрица порядка **2n**, элементы которой фо рмируются случайным образом и находятся в пределах от −10 до 10. Получить новую матрицу, переставляя ее блоки размера **n×n** в соответствии со схемой.

#include <iostream> //для работы с вводом/выводом

#include <time.h> //для работы со временем

**using** **namespace** std;//объявляем программе, что мы используем пространство имен std

**void** main()

{

**const** **int** n = 5;

**int** cols, rows;

**int** arr[2 \* n][2 \* n];    // 2-мерный масив для начальных

**int** rez[2 \* n][2 \* n];    // 2-мерный масив для итоговых

    srand(time(**NULL**)); // Инициализация генератора случайных чисел с использованием текущего времени

    // Заполнение массива случайными значениями от -10 до 10

**for** (**int** i = 0; i < 2 \* n; i++)

**for** (**int** j = 0; j < 2 \* n; j++)

            arr[i][j] = (rand() % 21 - 10);

    // Вывод начального массива

**for** (**int** i = 0; i < 2 \* n; i++) {

**for** (**int** j = 0; j < 2 \* n; j++) {

            // Вывод элементов массива с дополнительными пробелами для форматирования

**if** ((arr[i][j] < 10) && (arr[i][j] >= 0))

                cout << "  ";

**if** (((arr[i][j] < 0) && (arr[i][j] > -10)) || (arr[i][j] == 10))

                cout << ' ';

            cout << arr[i][j] << ' ';

        }

        cout << endl;

    }

    cout << endl;

    cout << endl;

    // Заполнение итогового массива в соответствии с заданными правилами

**for** (**int** i = 0; i < 2 \* n; i++)

**for** (**int** j = 0; j < 2 \* n; j++) {

            // Вычисление новых индексов для итогового массива

**if** (i < n)

                rows = i + n;

**else**

                rows = i - n;

**if** (j < n)

                cols = j + n;

**else**

                cols = j - n;

            // Присвоение значений из начального массива в итоговой

            rez[rows][cols] = arr[i][j];

        }

    // Вывод итогового массива

**for** (**int** i = 0; i < 2 \* n; i++) {

**for** (**int** j = 0; j < 2 \* n; j++) {

            // Вывод элементов итогового массива с дополнительными пробелами для форматирования

**if** ((rez[i][j] < 10) && (rez[i][j] >= 0))

                cout << "  ";

**if** (((rez[i][j] < 0) && (rez[i][j] > -10)) || (rez[i][j] == 10))

                cout << ' ';

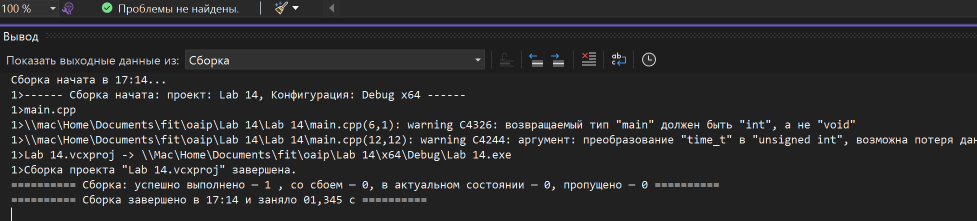
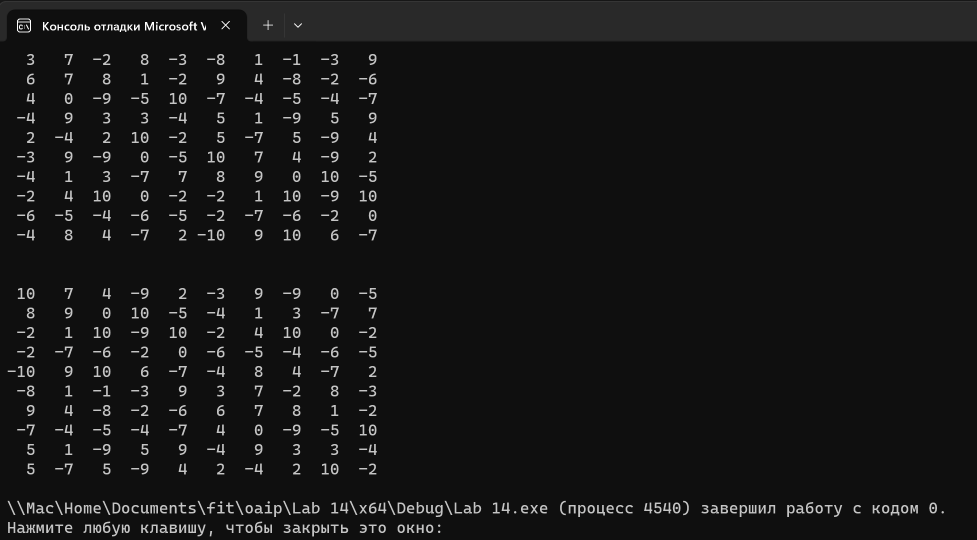
            cout << rez[i][j] << ' ';

        }

        cout << endl;

    }

}



2. Латинским квадратом порядка **n** называется квадратная таблица размером **nхn**, каждая строка и каждый столбец которой содержат все числа от 1 до **n**. Для заданного **n** в матрице **L(n, n**) построить латинский квадрат порядка **n**.

#include <iostream> //для работы с вводом/выводом

#include <time.h> //для работы со временем

**using** **namespace** std; //объявляем программе, что мы используем пространство имен std

**void** main()

{

**const** **int** n = 5; // размер масива

**int** arr[n][n];    // 2-мерный масив

    srand(time(**NULL**)); // Инициализация генератора случайных чисел с использованием текущего времени

    // Заполнение массива значениями в соответствии с условием

**for** (**int** i = 0; i < n; i++)

**for** (**int** j = 0; j < n; j++)

**if** (i + j >= n)

                arr[i][j] = i + j + 1 - n;

**else**

                arr[i][j] = i + j + 1;

    // Вывод массива

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

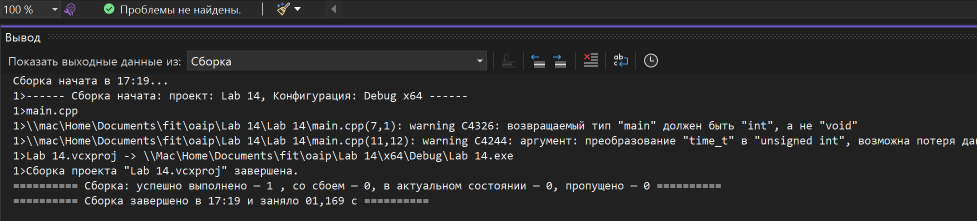
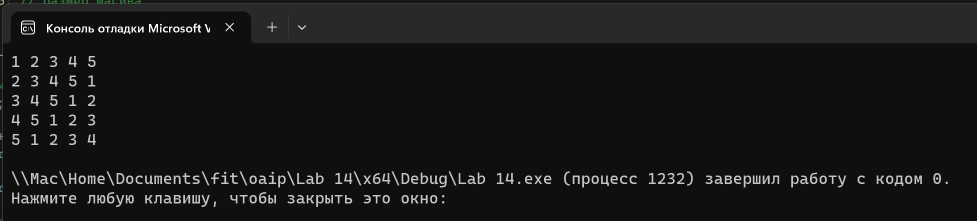
**for** (**int** j = 0; j < n; j++)

            cout << arr[i][j] << ' ';

        cout << endl;

    }

}



3. Путем перестановки элементов квадратной вещественной матрицы добиться того, чтобы ее максимальный элемент находился в левом верхнем углу, следующий по величине − в позиции (2, 2), следующий − в позиции (3, 3) и т. д., заполнив таким образом всю главную диагональ.

#include <iostream> //для работы с вводом/выводом

#include <ctime> //для работы со временем

#include <cstdlib> //для работы с памятью и случайными числами

#include <algorithm> //общие алгоритмы для работы с контейнерами

**using** **namespace** std; //объявляем программе, что мы используем пространство имен std

**int** main() {

    setlocale(LC\_CTYPE, "Russian"); //возможность работы с русским языком

    // Установка сида для генератора случайных чисел

    srand(time(**NULL**));

**const** **int** N = 100;

**double** a[N][N];

**int** n;

    // Ввод размерности матрицы

    cout << "Введите размерность матрицы: ";

    cin >> n;

    // Заполнение и вывод исходной матрицы

    cout << "Исходная матрица:\n";

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

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

            // Заполнение случайными числами от 0 до 99

            a[i][j] = rand() % 100;

            cout << a[i][j] << " ";

        }

        cout << endl;

    }

    // Сортировка диагонали в порядке убывания

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

**int** max\_i = i, max\_j = i;

**for** (**int** j = i; j < n; j++) {

**for** (**int** k = i; k < n; k++) {

                // Поиск максимального элемента в оставшейся части матрицы

**if** (a[j][k] > a[max\_i][max\_j]) {

                    max\_i = j;

                    max\_j = k;

                }

            }

        }

        // Перестановка максимального элемента на диагонали

**if** (max\_i != i || max\_j != i) {

            swap(a[i][i], a[max\_i][max\_j]);

        }

    }

    // Вывод преобразованной матрицы

    cout << "Преобразованная матрица:\n";

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

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

            cout << a[i][j] << " ";

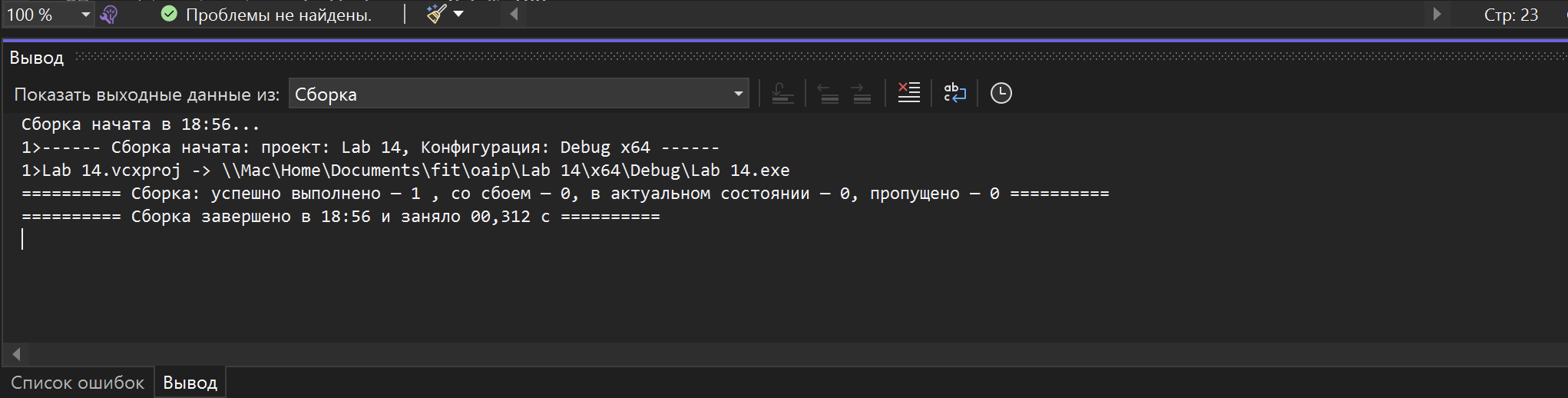
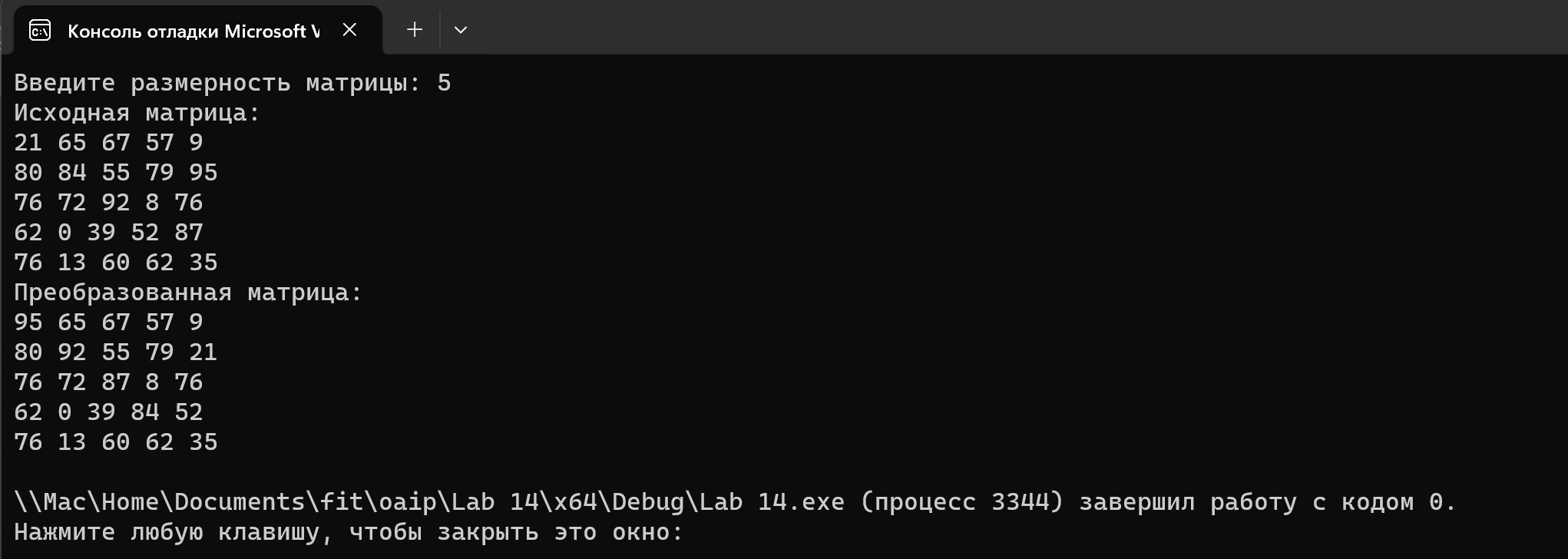
        }

        cout << endl;

    }

**return** 0;

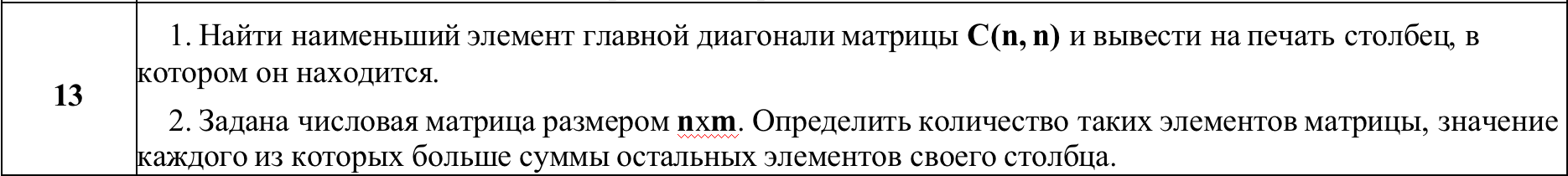
}



Другой вариант сделан по ошибке из-за невнимательности (не заметил доп задания).

Вариант 13

4. Выполнить задания из таблицы ниже, используя ***индексы*** для доступа к элементам массивов в первой программе и ***указатели*** − во второй программе.



1. Найти наименьший элемент главной диагонали матрицы **С(n, n)** и вывести на печать столбец, в котором он находится.

#include <iostream> //для работы с вводом/выводом

**using** **namespace** std; //объявляем программе, что мы используем пространство имен std

**int** main() {

    setlocale(LC\_CTYPE, "Russian"); //возможность работы с русским языком

**const** **int** n = 3; // Размер матрицы

**int** C[n][n] = {

        {5, -2, 3},

        {4, 9, 1},

        {-3, 6, -9}

    };

    // Поиск минимального элемента на главной диагонали и определение его столбца

**int** minElement = C[0][0];

**int** minCol = 0;

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

**if** (C[i][i] < minElement) {

            minElement = C[i][i];

            minCol = i;

        }

    }

    cout << "Наименьший элемент на главной диагонали: " << minElement << endl;

    cout << "Столбец, в котором находится минимальный элемент:\n";

    //выводим столбец в строку

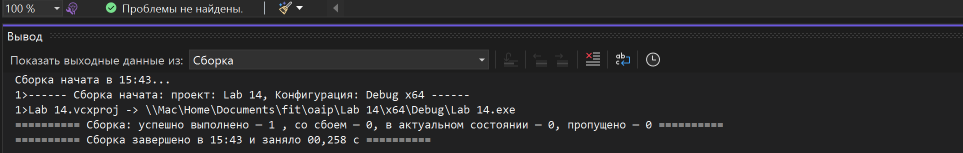
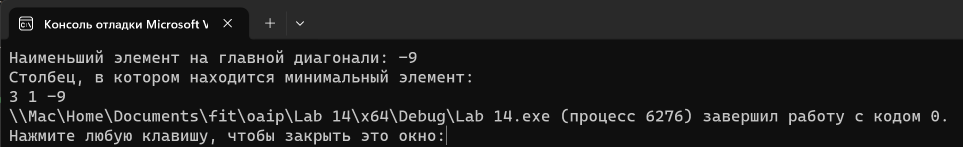
**for** (**int** i = 0; i < n; ++i) {

        cout << C[i][minCol] << " ";

    }

**return** 0;

}



2. Задана числовая матрица размером **n**x**m**.Определить количество таких элементов матрицы, значение каждого из которых больше суммы остальных элементов своего столбца.

#include <iostream> //для работы с вводом/выводом

**using** **namespace** std; //объявляем программе, что мы используем пространство имен std

**int** main() {

    setlocale(LC\_CTYPE, "Russian"); //возможность работы с русским языком

**const** **int** n = 4; // Количество строк

**const** **int** m = 4; // Количество столбцов

**int** A[n][m] = {

        {5, -2, 3, 1},

        {4, 9, 1, -3},

        {-3, 6, -9, 2},

        {0, 7, -1, 8}

    };

**int** countElements = 0; // Количество элементов, удовлетворяющих условию

    // Перебираем каждый столбец

**for** (**int** j = 0; j < m; ++j) {

        // Вычисляем сумму элементов столбца

**int** sum = 0;

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

            sum += \*(A[i] + [j]);

        }

        // Перебираем элементы столбца и проверяем условие

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

**if** (\*(A[i] + [j]) > sum - \*(A[i] + [j])) {

                // Элемент удовлетворяет условию

                countElements++;

            }

        }

    }

    // Вывод результатов

    cout << "Матрица:\n";

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

**for** (**int** j = 0; j < m; ++j) {

            cout << \*(A[i] + [j]) << " ";

        }

        cout << endl;

    }

    cout << "\nКоличество элементов, значение которых больше суммы остальных элементов своего столбца: " << countElements << endl;

**return** 0;

}

