Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

Лабораторная работа 15

По дисциплине «Основы алгоритмизации и программирования»

На тему «Динамическое выделение памяти»

Выполнил:

Студент 1 курса 9 группы

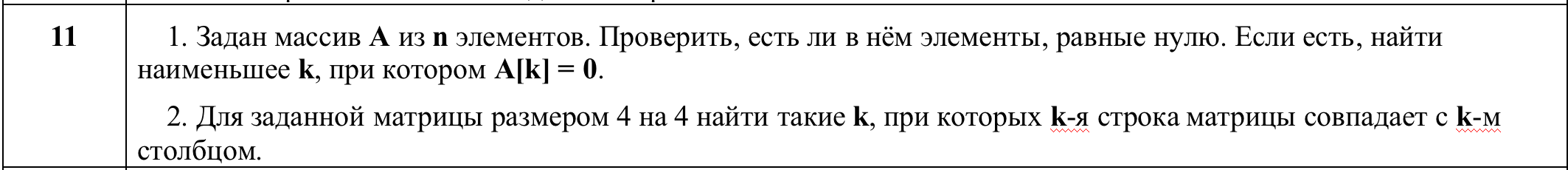
Павлович Ян Андреевич

Преподаватель: асс. Андронова М.В.

2023, Минск

Вариант 11

4. В соответствии со своим вариантом написать программы с использованием ***динамических*** массивов для условий задач из таблицы. Начальные размерности массивов ввести с клавиатуры. Первое задание выполнить с использованием ***функций*** для выделения динамической памяти на языке С, второе − с использованием операторов **new** и **delete**  на языке **С++**.



1. Задан массив **A** из **n** элементов. Проверить, есть ли в нём элементы, равные нулю. Если есть, найти наименьшее **k**, при котором **A[k] = 0**.

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

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

// Функция для выделения динамической памяти под массив

**int**\* createArray(**int** size) {

**return** **new** **int**[size];

}

// Функция для освобождения динамической памяти, выделенной под массив

**void** deleteArray(**int**\* arr) {

**delete**[] arr;

}

// Функция для нахождения наименьшего k, при котором A[k] = 0

**int** findZeroIndex(**const** **int**\* arr, **int** size) {

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

**if** (arr[i] == 0) {

**return** i;

        }

    }

**return** -1; // Если элементы, равные нулю, не найдены

}

**int** main() {

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

**int** n;

    // Ввод размерности массива

    cout << "Введите количество элементов в массиве: ";

    cin >> n;

    // Выделение динамической памяти под массив

**int**\* A = createArray(n);

    // Ввод элементов массива

    cout << "Введите элементы массива:\n";

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

        cout << "A[" << i << "]: ";

        cin >> A[i];

    }

    // Поиск индекса элемента, равного нулю

**int** zeroIndex = findZeroIndex(A, n);

    // Вывод результата

**if** (zeroIndex != -1) {

        cout << "Найден элемент, равный нулю, в позиции " << zeroIndex << ".\n";

    }

**else** {

        cout << "Элементы, равные нулю, не найдены.\n";

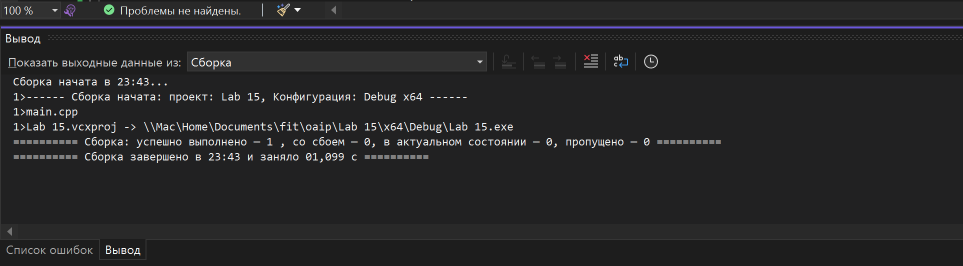
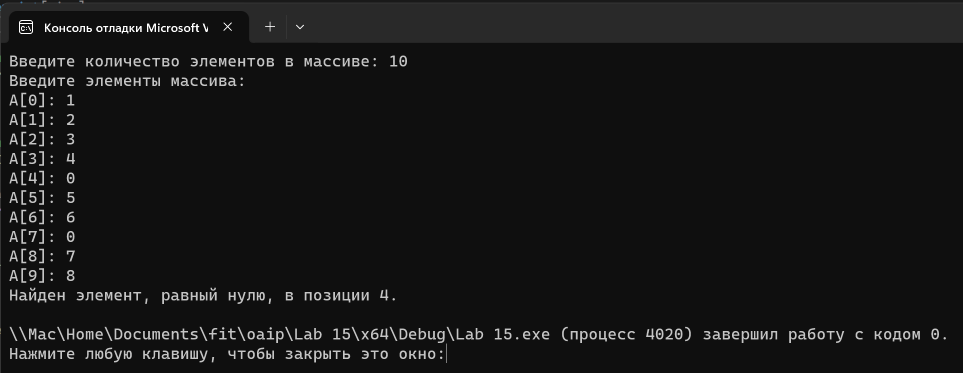
    }

    // Освобождение динамической памяти

    deleteArray(A);

**return** 0;

}



2. Для заданной матрицы размером 4 на 4 найти такие **k**, при которых **k**-я строка матрицы совпадает с **k**-м столбцом.

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

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

**int** main() {

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

**int** n;

    cout << "Введите размерность матрицы: ";

    cin >> n;

    // Выделение памяти под динамическую матрицу

**int**\*\* matrix = **new** **int**\* [n];

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

        matrix[i] = **new** **int**[n];

    }

    cout << "Введите элементы матрицы:" << endl;

    // Ввод элементов матрицы

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

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

            cin >> matrix[i][j];

        }

    }

    cout << "Результат:" << endl;

    // Поиск и вывод результатов

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

**if** (matrix[k][k] == matrix[k][0]) {

            cout << "k = " << k << " - строка совпадает с " << k << "-м столбцом" << endl;

        }

    }

    // Освобождение памяти

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

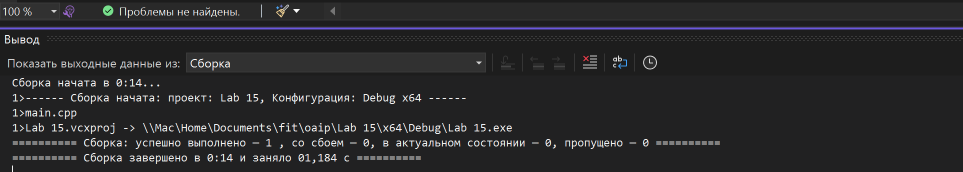
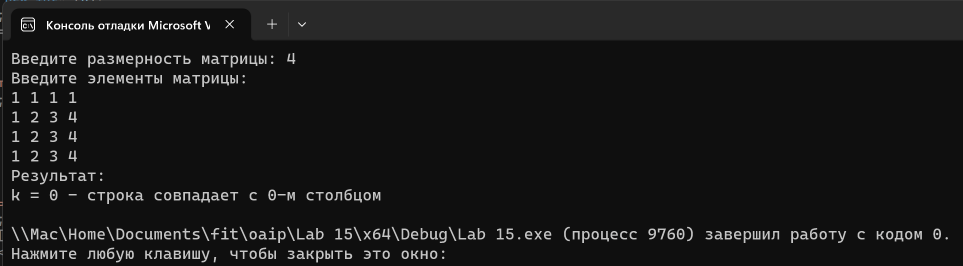
**delete**[] matrix[i];

    }

**delete**[] matrix;

**return** 0;

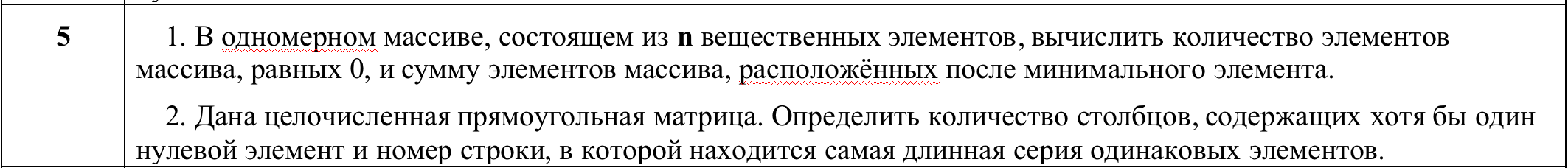
}



**Дополнительные задания**

Вариант 5

1. В соответствии со своим вариантом написать программы с использованием ***динамических*** массивов для условий задач из таблицы. Начальные размерности массивов ввести с клавиатуры. Первое задание выполнить с использованием ***функций*** для выделения динамической памяти на языке С, второе − с использованием операторов **new** и **delete**  на языке **С++**.



1. В одномерном массиве, состоящем из **n** вещественных элементов, вычислить количество элементов массива, равных 0, и сумму элементов массива, расположённых после минимального элемента.

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

#include<ctime>//  для работы с случайными числами

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

//Функция принимает минимум и максимум и возвращает случайное число в этих пределах

**int** randomNumberFunction(**int** rmn, **int** rmx);

**int** main() {

    setlocale(LC\_ALL, "ru");// для работы с русским языком

    srand((**unsigned**)time(**NULL**));//Функция делает числа случайными

**int** n, \* mainArray, сounterZero{}, sum{}, minElements{}, coordinatesMinElements{};// Инициализация указателя и переменных

    //Вводим количество элементов динамического массива

    cout << "Введите количество элементов массива: ";

    cin >> n;

    //Выделяем память для динамического массива с помощью функция языка C и проверяем выделилась ли она

**if** (!(mainArray = (**int**\*)malloc(n \* **sizeof**(**int**))))

    {

        cout << "Память не выделилась!" << endl;

**return** 1;

    }

    //Заполняем масиив случайными числами, считаем количество 0, вывыводим массив и находим минимальный элемент

    cout << "Динамический массив: ";

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

    {

        \*(mainArray + i) = randomNumberFunction(-10, 10);

**if** (\*(mainArray + i) < minElements || i == 0)

        {

            minElements = \*(mainArray + i);

            coordinatesMinElements = i;

        }

        cout << \*(mainArray + i) << ' ';

**if** (\*(mainArray + i) == 0)

        {

            ++сounterZero;

        }

    }

    //Считаем сумму чисел после минимального элемента

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

    {

        sum += \*(mainArray + i);

    }

    //Освобождение динамической памяти

    free(mainArray);

    //Выводим результат в консоль

    cout << "\nКоличество нулевых элементов: " << сounterZero << endl;

    cout << "Сумма элементов, расположенных после минимального элемента: " << sum << endl;

**return** 0;

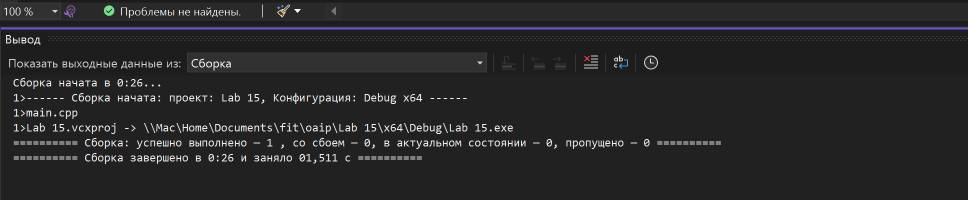
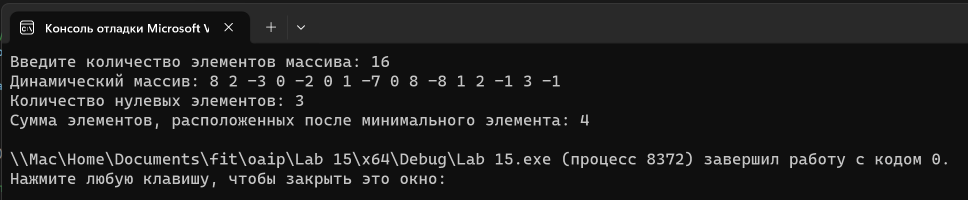
}

**int** randomNumberFunction(**int** rmn, **int** rmx) {

**int** number = (**int**)(((**double**)rand() / (**double**)RAND\_MAX) \* (rmx - rmn) + rmn);

**return** number;

}



2. Дана целочисленная прямоугольная матрица. Определить количество столбцов, содержащих хотя бы один нулевой элемент и номер строки, в которой находится самая длинная серия одинаковых элементов.

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

#include<ctime>//  для работы с случайными числами

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

//Функция принимает минимум и максимум и возвращает случайное число в этих пределах

**int** randomNumberFunction(**int** rmn, **int** rmx);

**int** main() {

    setlocale(LC\_ALL, "ru");// для работы с русским языком

    srand((**unsigned**)time(**NULL**));//Функция делает числа случайными

**int** line, columns, \*\* matrix, numberColumnsZero{}, counterIdentical{}, maxCounterIdentical{}, indexIdentical{};// Инициализация указателя и переменных

    //Вводим количество строк матрицы

    cout << "Введите количество строк матрицы: ";

    cin >> line;

    //Вводим количество колонок матрицы

    cout << "Введите количество колонок матрицы: ";

    cin >> columns;

    //Выделяем память под матрицу

    matrix = **new** **int**\* [line];

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

        matrix[i] = **new** **int**[columns];

    }

    //Заполняем матрицу случайными числами и выводим её

    cout << "Матрица: " << endl;

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

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

            matrix[i][j] = randomNumberFunction(0, 10);

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

        }

        cout << endl;

    }

    //Считаем количество колонок с нулевыми значениями и выводим результат

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

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

**if** (matrix[j][i] == 0) {

                ++numberColumnsZero;

**break**;

            }

        }

    }

    cout << "Количество столбцов, в которых есть нулевой элемент: " << numberColumnsZero << endl;

    //Определяем строку, в которой больше всего одинаковых элементов и выводим результат

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

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

            counterIdentical = 0;

**for** (**int** g = 0; g < columns; g++) {

**if** (matrix[i][j] == matrix[i][g]) {

                    ++counterIdentical;

                }

            }

**if** (counterIdentical > maxCounterIdentical) {

                maxCounterIdentical = counterIdentical;

                indexIdentical = i;

            }

        }

    }

    cout << "Номер строки, в которой находится самая длинная серия одинаковых элементов: " << indexIdentical + 1 << endl;

    //Освобождение памяти

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

**delete**[] matrix[i];

    }

**delete**[] matrix;

**return** 0;//Главная функция возвращает 0 при успешном выполнении

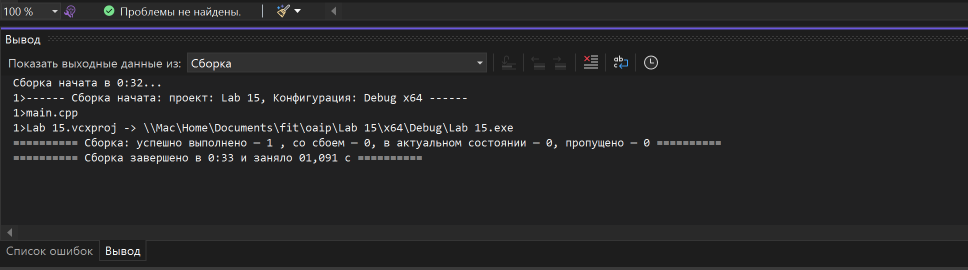
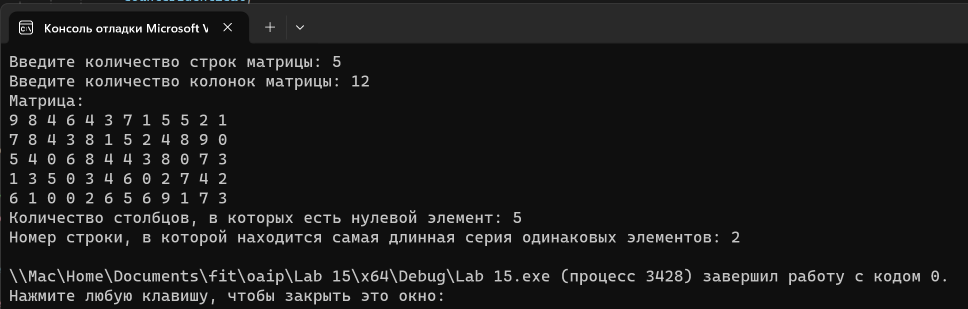
}

**int** randomNumberFunction(**int** rmn, **int** rmx) {

**int** number = (**int**)(((**double**)rand() / (**double**)RAND\_MAX) \* (rmx - rmn) + rmn);

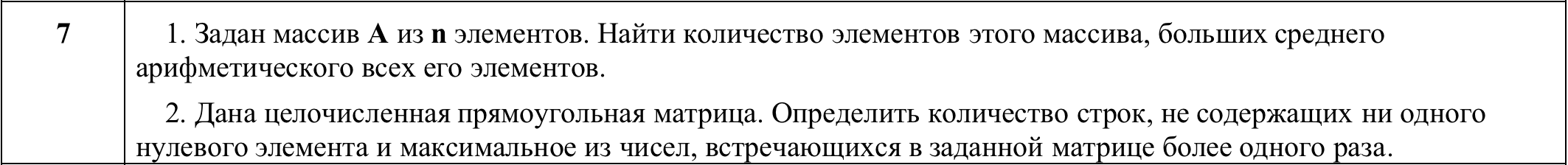
**return** number;

}



Вариант 7

4. В соответствии со своим вариантом написать программы с использованием ***динамических*** массивов для условий задач из таблицы. Начальные размерности массивов ввести с клавиатуры. Первое задание выполнить с использованием ***функций*** для выделения динамической памяти на языке С, второе − с использованием операторов **new** и **delete**  на языке **С++**.



1. Задан массив **A** из **n** элементов. Найти количество элементов этого массива, больших среднего арифметического всех его элементов.

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

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

**int** main() {

    setlocale(LC\_ALL, "Russian");// для работы с русским языком

**float** avr = 0;

**int** n;

    cout << "Введите резмер массива: ";

    cin >> n; //создаем переменные

**int**\* a = **new** **int**[n]; //создание массива

    cout << "Введите массив: ";

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

        cin >> a[i]; //ввод массива

        avr += a[i]; //нахождение суммы элементов

    }

    avr = avr / n; //нахождение среднего арифметического

**int** count = 0;

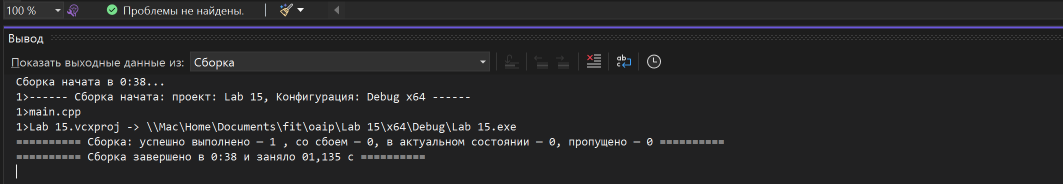
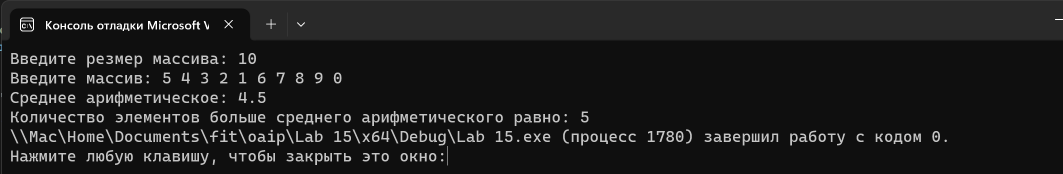
**for** (**int** i = 0; i < n; i++) {

        a[i] > avr ? count++ : count = count; //подсчет элементов

    }

    cout << "Среднее арифметическое: " << avr << endl << "Количество элементов больше среднего арифметического равно: " << count;

}



1. Дана целочисленная прямоугольная матрица. Определить количество строк, не содержащих ни одного нулевого элемента и максимальное из чисел, встречающихся в заданной матрице более одного раза.

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

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

**int** main() {

    setlocale(LC\_ALL, "Russian");// для работы с русским языком

**int**\*\* a;

**int** n, m;

    cout << "Введите размер матрицы: "; cin >> n >> m;

    a = **new** **int**\* [n]; //создание переменных

    cout << "Введите матрицу: ";

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

        a[i] = **new** **int**[m]; //отделение памяти

    }

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

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

            cin >> a[i][j]; //ввод матрицы

        }

    }

**int** max = a[0][0];

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

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

**if** (a[i][j] > max) {

                max = a[i][j]; //поиск максимального элемента в матрице

            }

        }

    }

**int** cnt = 0;

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

**bool** status = **true**; //проверка элементов строк матрицы

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

**if** (a[i][j] == 0) {

                status = **false**;

**break**;

            }

        }

**if** (status) {

            cnt++; //подсчет

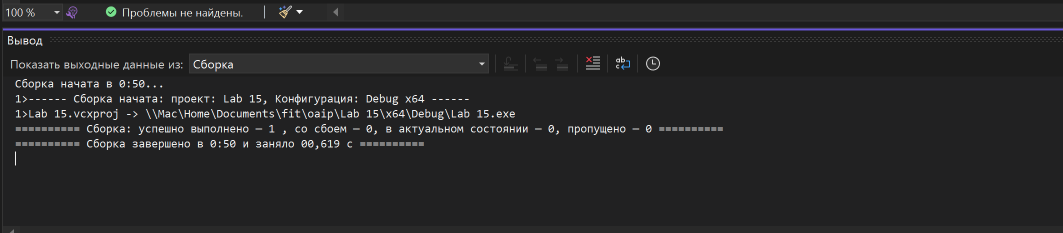
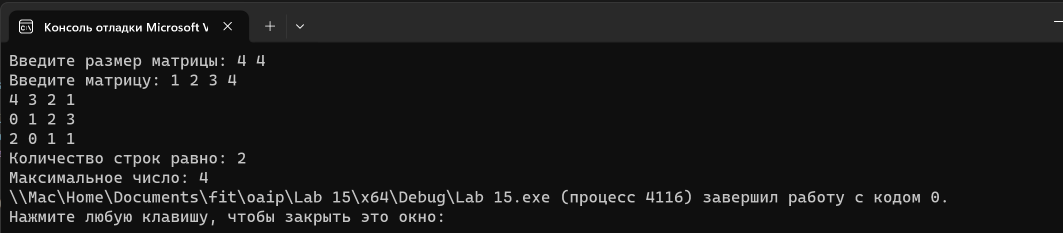
        }

    }

    cout << "Количество строк равно: " << cnt;

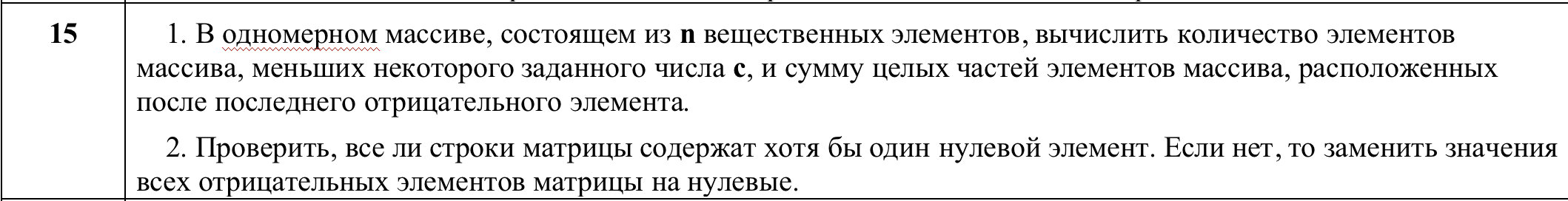
    cout << "\nМаксимальное число: " << max;

}



Вариант 15

4. В соответствии со своим вариантом написать программы с использованием ***динамических*** массивов для условий задач из таблицы. Начальные размерности массивов ввести с клавиатуры. Первое задание выполнить с использованием ***функций*** для выделения динамической памяти на языке С, второе − с использованием операторов **new** и **delete**  на языке **С++**.



1. В одномерном массиве, состоящем из **n** вещественных элементов, вычислить количество элементов массива, меньших некоторого заданного числа **с**, и сумму целых частей элементов массива, расположенных после последнего отрицательного элемента.

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

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

**int** main() {

    setlocale(LC\_ALL, "Russian");// для работы с русским языком

**float** avr = 0;

**int** n, c;

    cout << "Введите c: "; cin >> c;

    cout << "Введите резмер массива: ";

    cin >> n;

**float**\* a = **new** **float**[n]; //создание массива

    cout << "Введите массив: ";

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

        cin >> a[i]; //ввод массива

**if** (a[i] < c) {

            avr++;

        }

    }

**int** left = 0;

**for** (**int** i = n - 1; i > 0; i--) {

**if** (a[i] < 0) {

            left = i;//номер последнего отрицательного элемента

**break**;

        }

    }

    cout << "Кол-во элементов меньше с: " << avr << endl;

**int** sum = 0;

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

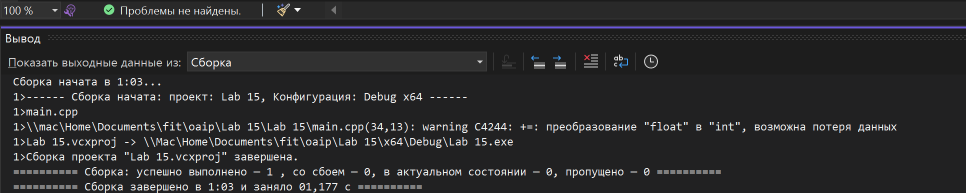
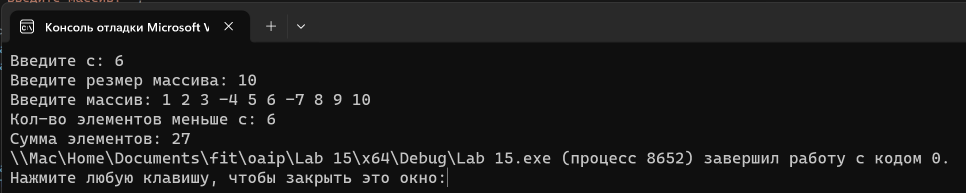
        sum += a[i]; //находим сумму

    }

    cout << "Сумма элементов: " << sum; //вывод

**return** 0;

}



2. Проверить, все ли строки матрицы содержат хотя бы один нулевой элемент. Если нет, то заменить значения всех отрицательных элементов матрицы на нулевые.

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

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

**int** main() {

    setlocale(LC\_ALL, "Russian");// для работы с русским языком

**int**\*\* a;

**int** n, m;

    cout << "Введите размер матрицы: "; cin >> n >> m;

    cout << "Введите матрицу: " << endl;

    a = **new** **int**\* [n]; //создание переменных

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

        a[i] = **new** **int**[m]; //создание матрицы

    }

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

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

            cin >> a[i][j]; //ввод матрицы

        }

    }

**int** cnt = 0;

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

**bool** status = **false**; //проверка элементов строк матрицы

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

**if** (a[i][j] == 0) {

                status = **true**;

**break**;

            }

        }

**if** (status) {

            cnt += 1; //сумма кол-ва строк с 0

        }

    }

**if** (cnt == n) {//проверка

        cout << "Все строки содержат нулевой элемент равный 0"; //вывод 1

    } **else** {

        cout << "Не все строки содержат 0" << endl << "Новая матрица: " << endl; //вывод 2

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

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

                a[i][j] > 0 ? cout << a[i][j] << " " : cout << 0 << " "; //проверка

            }

            cout << endl;

        }

    }

**return** 0;

}

