Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

Лабораторная работа 17

По дисциплине «Основы алгоритмизации и программирования»

на тему «**Указатели и ссылки при работе с функциями**»

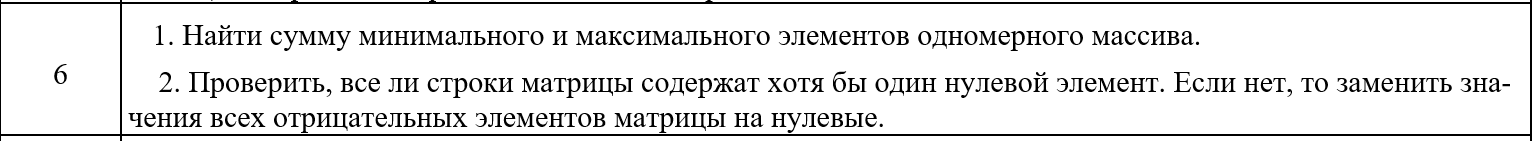
Выполнила:

Студент(ка) 1 курса 7 группы

Гриценко Анна Александровна

Преподаватель: асс. Андронова М.В.

2023, Минск



**Задание 1**

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

using namespace std;

void findMinMaxSum(const int\* arr, int size, int& min, int& max) {

// Инициализация min и max начальными значениями

min = arr[0];

max = arr[0];

// Поиск минимального и максимального элементов в массиве

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

if (arr[i] < min) {

min = arr[i];

}

if (arr[i] > max) {

max = arr[i];

}

}

}

int main() {

setlocale(LC\_ALL, "Russian");//функция для корректного отображения русского языка

int size;

// Ввод размера массива

cout << "Введите размер массива: ";

cin >> size;

// Выделение динамической памяти под массив

int\* array = new int[size];

// Ввод элементов массива

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

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

cout << "Элемент " << i + 1 << ": ";

cin >> array[i];

}

// Переменные для хранения минимального и максимального элементов

int minElement, maxElement;

// Вызов функции для нахождения минимального и максимального элементов

findMinMaxSum(array, size, minElement, maxElement);

// Вывод результатов

cout << "Минимальный элемент: " << minElement << endl;

cout << "Максимальный элемент: " << maxElement << endl;

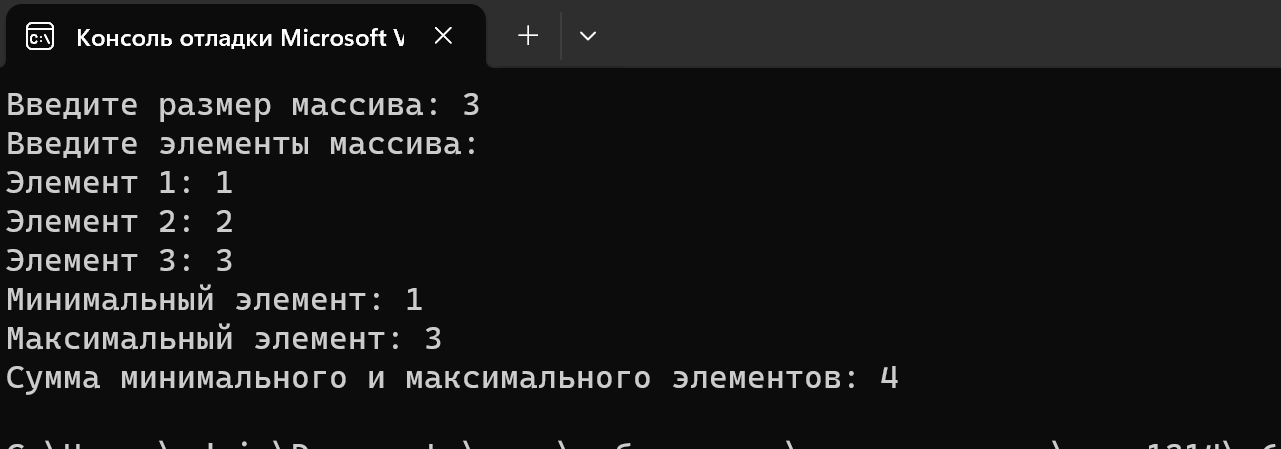
cout << "Сумма минимального и максимального элементов: " << minElement + maxElement << endl;

// Освобождение выделенной памяти

delete[] array;

return 0;

}



**Задание 2**

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

using namespace std;

// Функция для проверки, все ли строки матрицы содержат хотя бы один нулевой элемент

bool hasZeroInAllRows(const int\* matrix, int rows, int cols) {

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

bool hasZero = false;

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

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

hasZero = true;

break;

}

}

if (!hasZero) {

return false; // Если хотя бы в одной строке нет нуля, возвращаем false

}

}

return true;

}

// Функция для замены отрицательных элементов матрицы на нули

void replaceNegativesWithZero(int\* matrix, int rows, int cols) {

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

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

if (matrix[i \* cols + j] < 0) {

matrix[i \* cols + j] = 0;

}

}

}

}

// Функция для вывода матрицы на экран

void printMatrix(const int\* matrix, int rows, int cols) {

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

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

cout << matrix[i \* cols + j] << " ";

}

cout << endl;

}

}

int main() {

setlocale(LC\_ALL, "Russian");//функция для корректного отображения русского языка

int rows, cols;

// Ввод размеров матрицы

cout << "Введите количество строк матрицы: ";

cin >> rows;

cout << "Введите количество столбцов матрицы: ";

cin >> cols;

// Выделение динамической памяти под матрицу

int\* matrix = new int[rows \* cols];

// Ввод элементов матрицы

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

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

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

cout << "Элемент [" << i + 1 << "][" << j + 1 << "]: ";

cin >> matrix[i \* cols + j];

}

}

// Проверка, все ли строки матрицы содержат хотя бы один нулевой элемент

if (!hasZeroInAllRows(matrix, rows, cols)) {

// Если не все строки содержат хотя бы один нулевой элемент, заменяем отрицательные элементы на 0

replaceNegativesWithZero(matrix, rows, cols);

cout << "Заменены отрицательные элементы на 0:" << endl;

printMatrix(matrix, rows, cols);

}

else {

cout << "Все строки содержат хотя бы один нулевой элемент." <<endl;

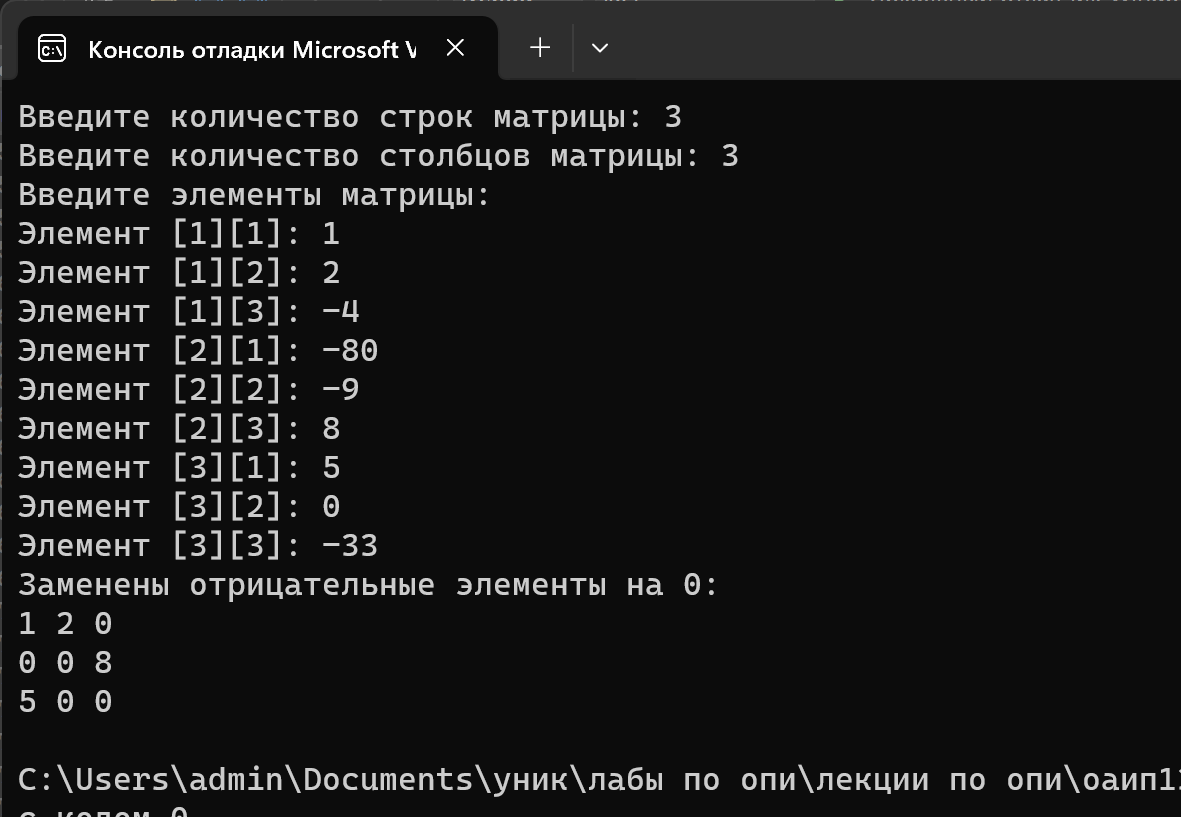
}

// Освобождение выделенной памяти

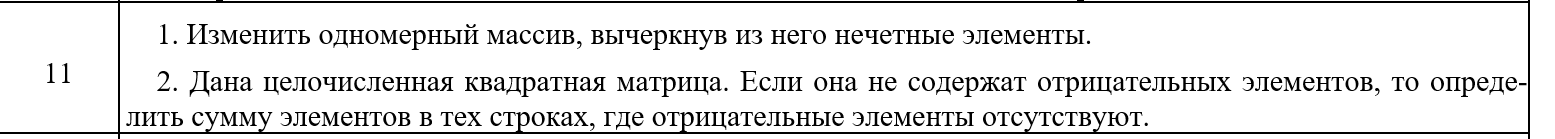
delete[] matrix;

return 0;

}



**Дополнительные задания**

****

**Задание 1**

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

using namespace std;

void removeOddElements(int\*& arr, int& size) {

int newSize = 0;

// Подсчет количества четных элементов

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

if (arr[i] % 2 == 0) {

++newSize;

}

}

// Создание нового массива без нечетных элементов

int\* newArr = new int[newSize];

int index = 0;

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

if (arr[i] % 2 == 0) {

newArr[index] = arr[i];

++index;

}

}

// Освобождение памяти старого массива

delete[] arr;

// Переназначение указателя на новый массив и обновление размера

arr = newArr;

size = newSize;

}

int main() {

setlocale(LC\_ALL, "Russian");//функция для корректного отображения русского языка

int size;

// Ввод размера массива

cout << "Введите размер массива: ";

cin >> size;

// Выделение динамической памяти под массив

int\* array = new int[size];

// Ввод элементов массива

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

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

cout << "Элемент " << i + 1 << ": ";

cin >> array[i];

}

// Вызов функции для удаления нечетных элементов

removeOddElements(array, size);

// Вывод результата

cout << "Измененный массив без нечетных элементов:" << endl;

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

cout << array[i] << " ";

}

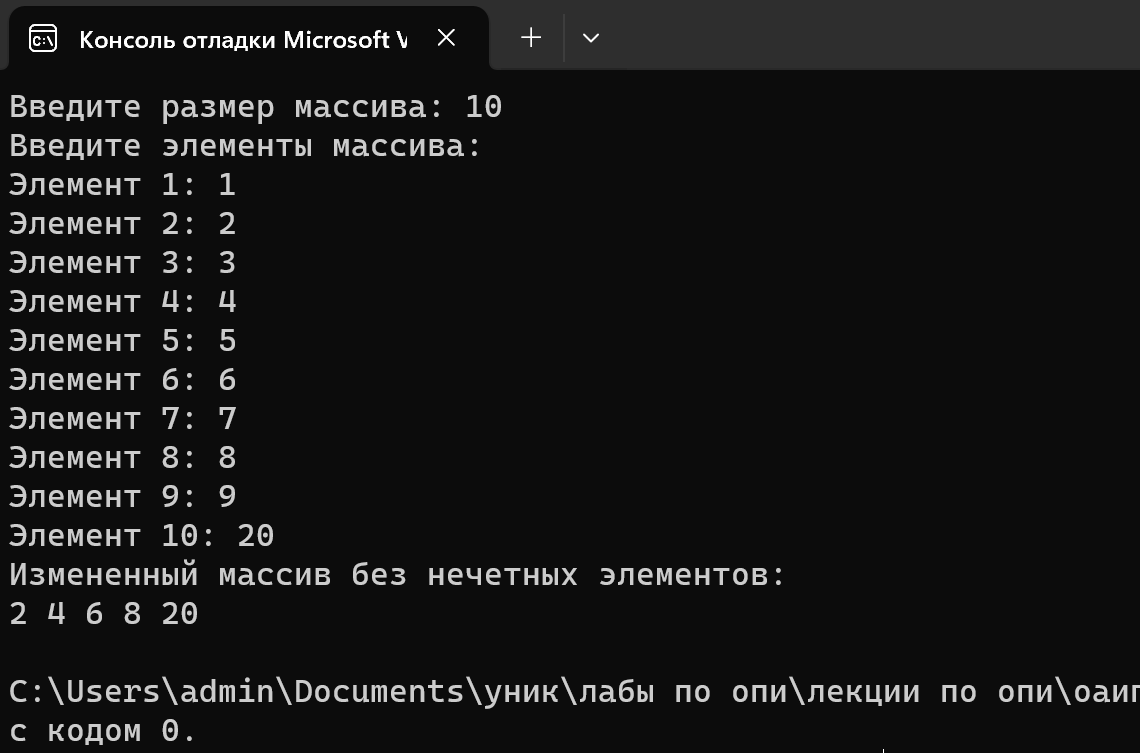
cout << endl;

// Освобождение выделенной памяти

delete[] array;

return 0;

}

****

**Задание 2**

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

using namespace std;

bool hasNegativeElement(int\*\* matrix, int n) {

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

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

if (matrix[i][j] < 0) {

return true;

}

}

}

return false;

}

void sum(int\*\* matrix, int n) {

// Проверка, содержит ли матрица отрицательные элементы

if (hasNegativeElement(matrix, n)) {

cout << "Матрица содержит отрицательные элементы." << endl;

return;

}

// Сумма элементов в каждой строке

int\* rowSums = new int[n]();

// Вычисление суммы элементов в каждой строке

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

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

rowSums[i] += matrix[i][j];

}

}

// Вывод сумм элементов в строках, где отрицательные элементы отсутствуют

cout << "Суммы элементов в строках без отрицательных элементов:" << endl;

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

cout << "Строка " << i + 1 << ": " << rowSums[i] <<endl;

}

// Освобождение памяти

delete[] rowSums;

}

int main() {

setlocale(LC\_ALL, "Russian");//функция для корректного отображения русского языка

int n;

// Ввод размера матрицы

cout << "Введите размер матрицы (n): ";

cin >> n;

// Выделение динамической памяти под матрицу

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

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

matrix[i] = new int[n];

}

// Ввод элементов матрицы

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

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

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

cout << "Элемент [" << i + 1 << "][" << j + 1 << "]: ";

cin >> matrix[i][j];

}

}

// Вызов функции для определения сумм элементов в строках без отрицательных элементов

sum(matrix, n);

// Освобождение памяти

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

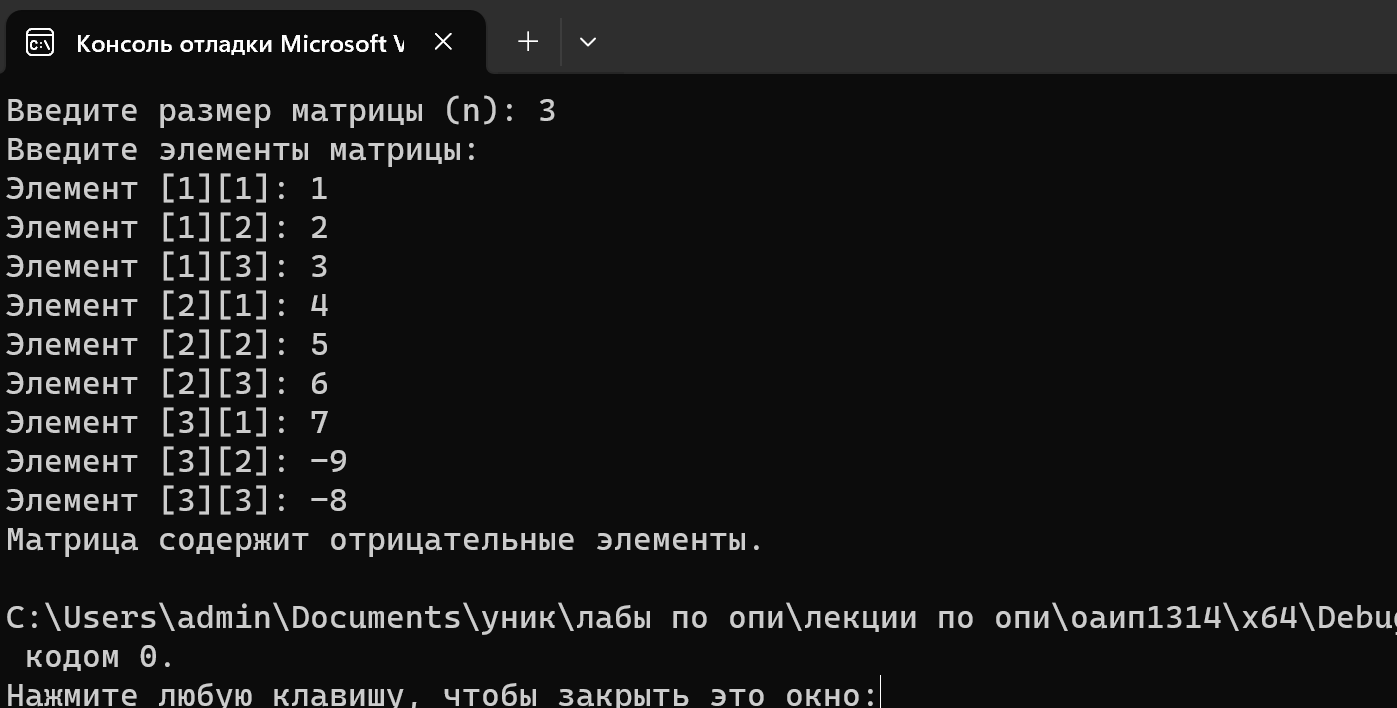
delete[] matrix[i];

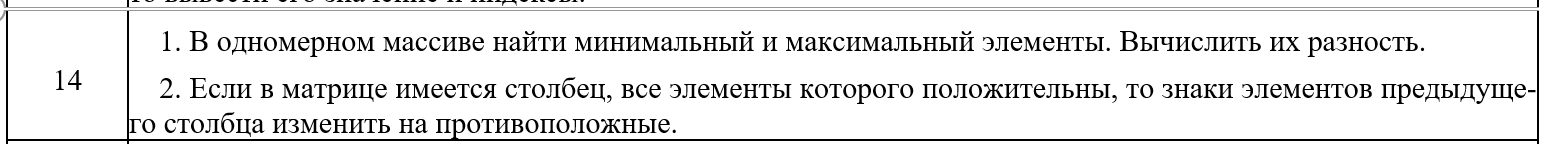
}

delete[] matrix;

return 0;

}

****

****

**Задание 1**

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

#include <climits>

using namespace std;

// Функция пользователя для нахождения минимального элемента в массиве

int findMin(const int array[], int size) {

int minElement = INT\_MAX;

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

if (array[i] < minElement) {

minElement = array[i];

}

}

return minElement;

}

// Функция пользователя для нахождения максимального элемента в массиве

int findMax(const int array[], int size) {

int maxElement = INT\_MIN;

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

if (array[i] > maxElement) {

maxElement = array[i];

}

}

return maxElement;

}

// Функция пользователя для вычисления разности между минимальным и максимальным элементами

int Difference(int minElement, int maxElement) {

return maxElement - minElement;

}

int main() {

setlocale(LC\_ALL, "Rus");//функция для корректного отображения русского языка

const int size = 10; // Задаем размер массива

int array[size]; // Объявляем массив

// Заполняем массив случайными значениями (можете ввести значения с клавиатуры или считать из файла)

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

cout << "Введите элемент массива[" << i << "]: ";

cin >> array[i];

}

// Вызываем функции для нахождения минимального и максимального элементов

int minElement = findMin(array, size);

int maxElement = findMax(array, size);

// Вычисляем разность между минимальным и максимальным элементами

int difference =Difference(minElement, maxElement);

// Вывод результаты

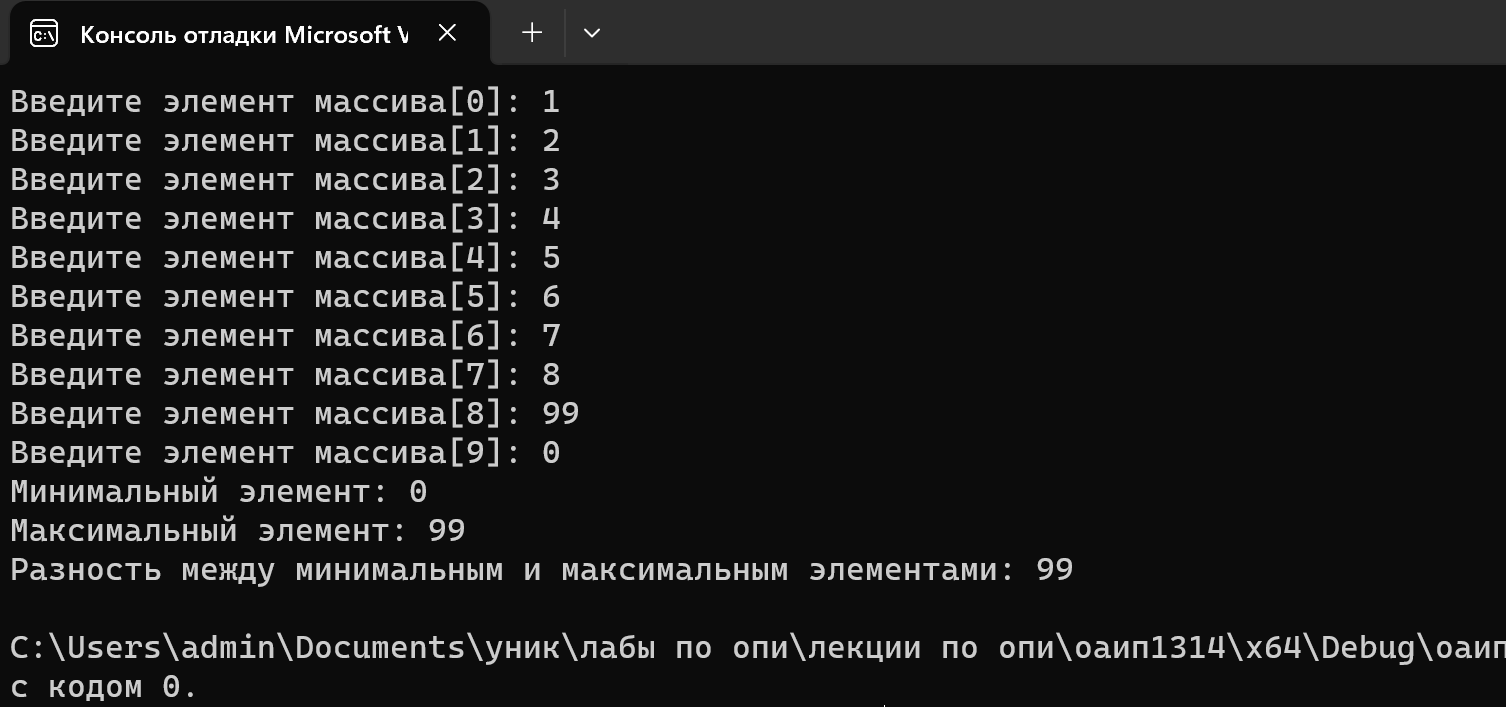
cout << "Минимальный элемент: " << minElement << endl;

cout << "Максимальный элемент: " << maxElement << endl;

cout << "Разность между минимальным и максимальным элементами: " << difference <<endl;

return 0;

}



**Задание 2**

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

#include <climits>

using namespace std;

const int rows = 3; //Указываем количество строк

const int columns = 4; //Указываем количество столбцов

// Функция пользователя для проверки, все ли элементы столбцаявляются положительны

bool isColumnPositive(const int array[rows][columns], int col) {

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

if (array[i][col] <= 0) {

return false;

}

}

return true;

}

// Функция для изменения знаков элементов предыдущего столбца

void changeSignsPreviousColumn(int matrix[rows][columns], int col) {

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

matrix[i][col - 1] \*= -1;

}

}

// Функция для вывода матрицы

void printMatrix(const int matrix[rows][columns]) {

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

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

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

}

cout << endl;

}

}

int main() {

setlocale(LC\_ALL, "Rus");//функция для корректного отображения русского языка

int matrix[rows][columns]; // Объявляем матрицу

// Заполняем матрицу (можете ввести значения с клавиатуры или считать из файла)

cout << "Введите элементы матрицы " << rows << "x" << columns << ":\n";

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

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

cin >> matrix[i][j];

}

}

// Проверяем каждый столбец на положительность и меняем знаки предыдущего столбца

for (int j = 1; j < columns; ++j) {

if (isColumnPositive(matrix, j)) {

changeSignsPreviousColumn(matrix, j);

break; // Если найден положительный столбец, прекращаем поиск

}

}

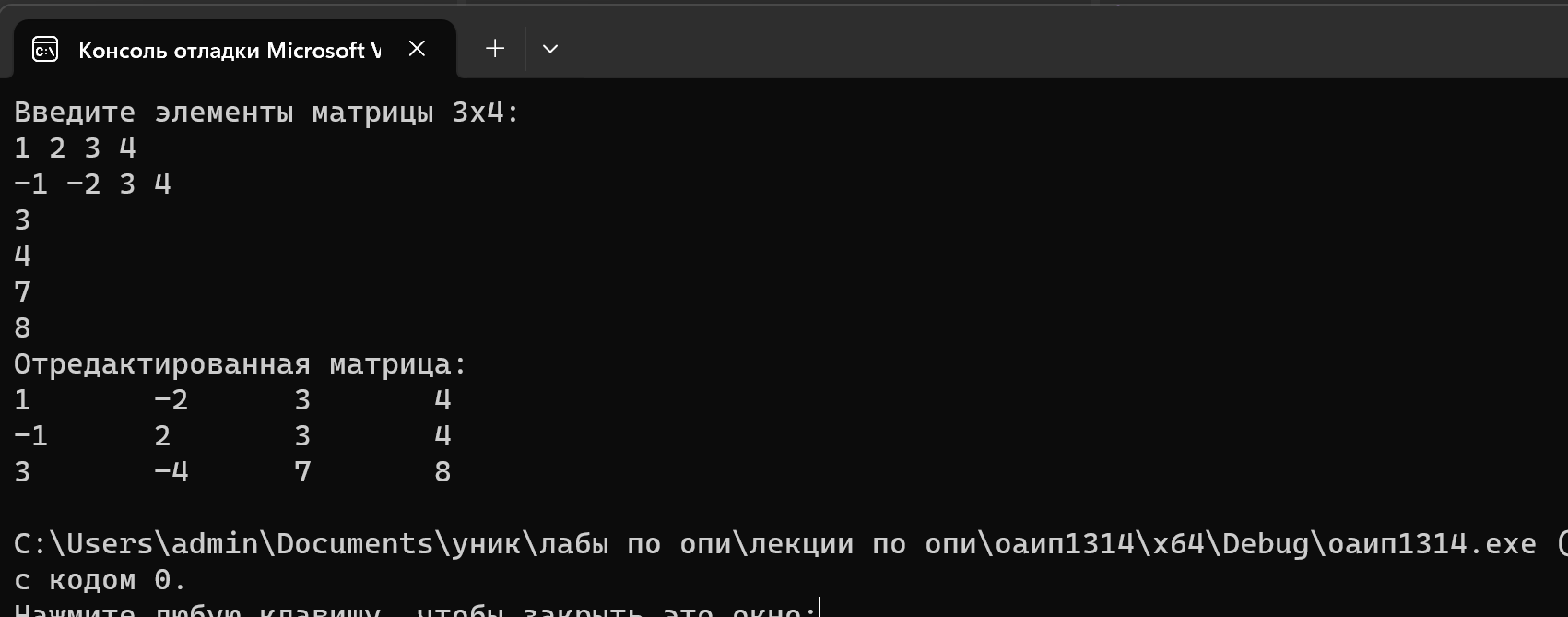
// Вывод результаты

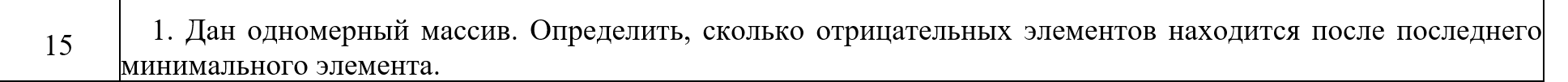
cout << "Исходная матрица:\n";

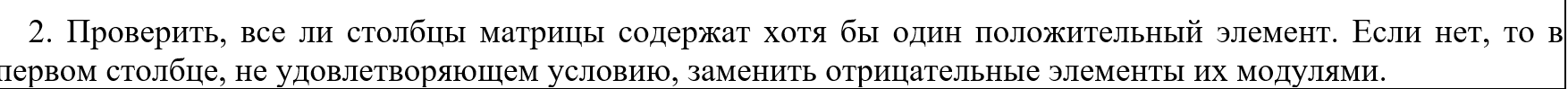
printMatrix(matrix);

return 0;

}

****

****

****

**Задание 1**

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

#include <climits>

using namespace std;

// Функция пользователя для поиска индекса последнего минимального элемента

int findLastMinIndex(const int array[], int size) {

if (size <= 0) {

return -1; // Возвращаем -1, если размер массива некорректен

}

int minElement = array[size - 1];

int lastIndex = size - 1;

for (int i = size - 2; i >= 0; --i) {

if (array[i] < minElement) {

minElement = array[i];

lastIndex = i;

}

}

return lastIndex;

}

// Функция для подсчета отрицательных элементов после последнего минимального

int countMin(const int array[], int size) {

int lastMinIndex = findLastMinIndex(array, size);

if (lastMinIndex == -1) {

return 0; // Если не найден минимальный элемент, возвращаем 0

}

int count = 0;

for (int i = lastMinIndex + 1; i < size; ++i) {

if (array[i] < 0) {

++count;

}

}

return count;

}

int main() {

setlocale(LC\_ALL, "Rus");//функция для корректного отображения русского языка

const int size = 10; // Задаем размер массива

int array[size]; // Объявляем массив

// Заполняем массив случайными значениями

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

cout << "Введите элемент массива[" << i << "]: ";

cin >> array[i];

}

// Вызываем функцию для подсчета отрицательных элементов после последнего минимального

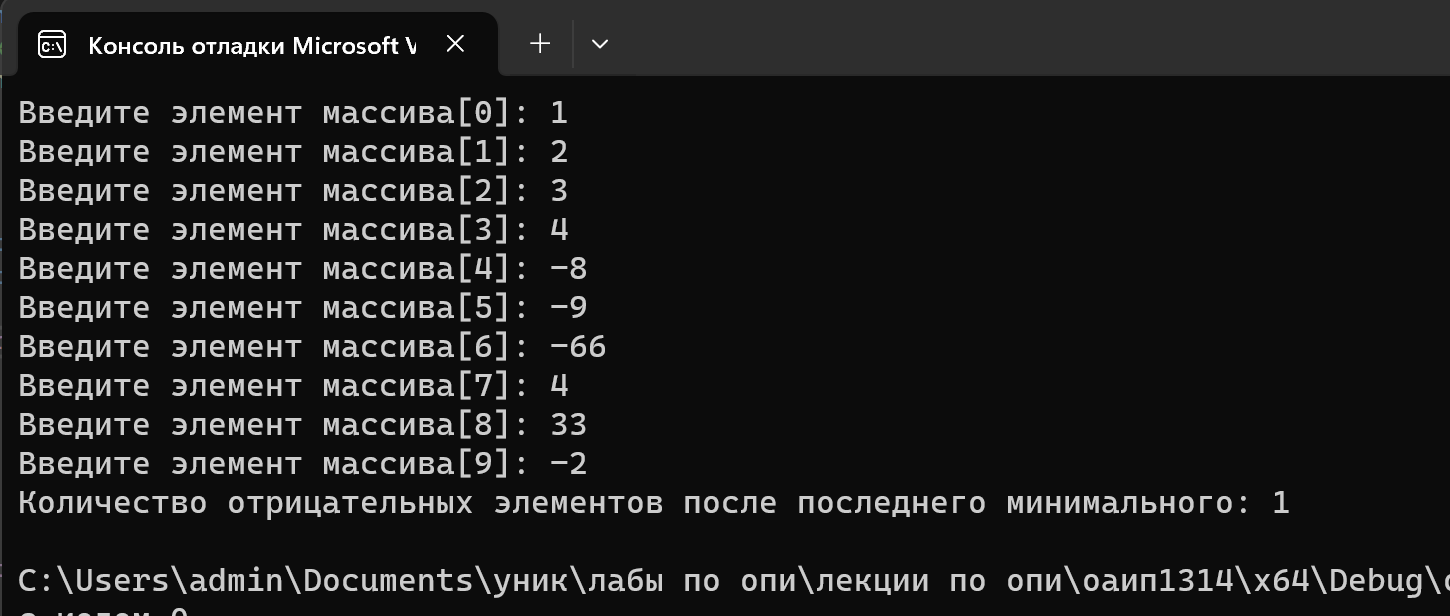
int countNegatives = countMin(array, size);

// Выводим результат

cout << "Количество отрицательных элементов после последнего минимального: " << countNegatives << endl;

return 0;

}

****

**Задание 2**

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

#include <climits>

using namespace std;

const int rows = 3; // Задаем количество строк

const int cols = 4; // Задаем количество столбцов

// Функция для проверки, содержит ли столбец хотя бы один положительный элемент

bool ColumnPositive(const int matrix[rows][cols], int col) {

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

if (matrix[i][col] > 0) {

return true;

}

}

return false;

}

// Функция для замены отрицательных элементов в столбце на их модули

void replace(int matrix[rows][cols], int col) {

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

if (matrix[i][col] < 0) {

matrix[i][col] = -matrix[i][col];

}

}

}

// Функция для вывода матрицы

void printMatrix(const int matrix[rows][cols]) {

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

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

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

}

cout <<endl;

}

}

int main() {

setlocale(LC\_ALL, "Rus");//функция для корректного отображения русского языка

int matrix[rows][cols]; // Объявляем матрицу

// Заполняем матрицу

cout << "Введите элементы матрицы " << rows << "x" << cols << ":\n";

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

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

cin >> matrix[i][j];

}

}

// Проверяем каждый столбец на наличие хотя бы одного положительного элемента

// и при необходимости заменяем отрицательные элементы в первом столбце

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

if (!ColumnPositive(matrix, j)) {

replace(matrix, 0);

break; // Если найден столбец без положительных элементов, прекращаем поиск

}

}

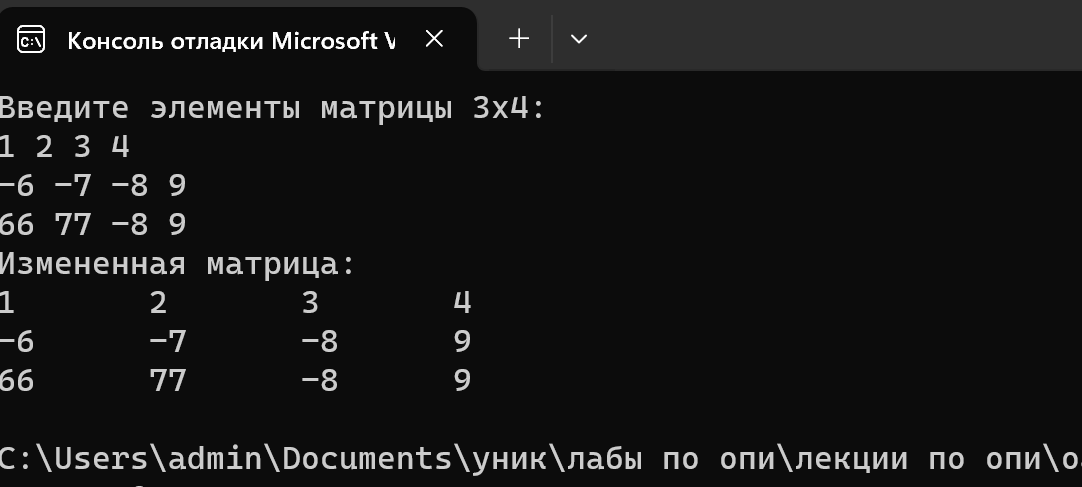
// Выводим результаты

cout << "Измененная матрица:\n";

printMatrix(matrix);

return 0;

}

****