**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»

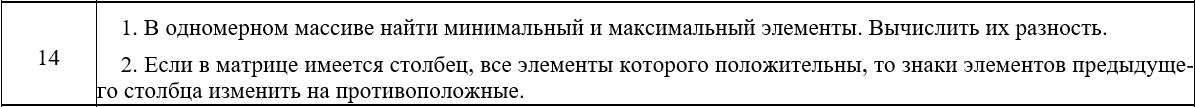
Выполнила:

Студентка 1 курса 7 группы

Шинкевич Марина Дмитриевна

Преподаватель: асс. Андронова М.В.

2023, Минск



#include <iostream>

#include <limits>

using namespace std;

//Объявление функции find\_Difference, которая принимает указатель на массив целых чисел arr, размер массива size и ссылки на переменные minEl, maxEl и diff.

void find\_Difference(int\* arr, int size, int& minEl, int& maxEl, int& diff) {

minEl = numeric\_limits<int>::max(); //Инициализация переменной minEl максимальным значением типа int.

maxEl = numeric\_limits<int>::min(); // Инициализация переменной maxEl минимальным значением типа int.

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

if (arr[i] < minEl) {//Если текущий элемент массива меньше значения minEl, то обновить значение minEl.

minEl= arr[i];

}

if (arr[i] > maxEl) {//Если текущий элемент массива больше значения maxEl, то обновить значение maxEl.

maxEl = arr[i];

}

}

diff = maxEl - minEl;// Вычисление разности между максимальным и минимальным элементами массива

}

int main() {

setlocale(LC\_ALL, "Rus");

int size;

cout << "Введите размер массива: ";

cin >> size;

int\* dynamicArray = new int[size];//создание динамического массива

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

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

cout << i + 1 << ": ";

cin >> dynamicArray[i];

}

int minEl, maxEl, diff;

find\_Difference(dynamicArray, size, minEl, maxEl, diff);

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

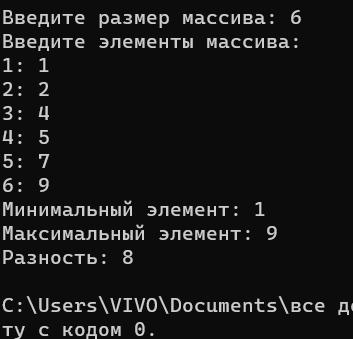
cout << "Максимальный элемент: " << maxEl << endl;

cout << "Разность: " << diff << endl;

delete[] dynamicArray;//освобождение памяти, выделенной под динамический массив.

return 0;

}



#include <iostream>

#include <iomanip>

using namespace std;

void Matrix\_1(int\*\* matrix, int rows, int colums) {//определение функции Matrix\_1, которая принимает двумерный массив (матрицу) matrix, количество строк rows и количество столбцов colums.

for (int colum = 1; colum < colums; ++colum) {//цикл для перебора столбцов матрицы

bool Positive\_elem = true;//переменной Positive\_elem, которая указывает, что все элементы в текущем столбце положительные. Изначально предполагается, что это так.

for (int row = 0; row < rows; ++row) {//Цикл, который перебирает все строки в текущем столбце

if (matrix[row][colum] <= 0) {//Проверка, является ли текущий элемент в столбце отрицательным или нулевым.

Positive\_elem = false;// Если элемент отрицательный или нулевой, переменная Positive\_elem устанавливается в false, и цикл прерывается.

break;

}

}

if (Positive\_elem) {//Если все элементы в текущем столбце положительные

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

matrix[row][colum - 1] \*= -1;//Умножение каждого элемента предыдущего столбца на -1, чтобы изменить знак.

}

}

}

}

void Matrix\_2(int\*\* matrix, int rows, int colums) {//функция Matrix\_2 которая принимает двумерный массив (матрицу) matrix, количество строк rows и количество столбцов colums.

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

for (int colum = 0; colum < colums; ++colum) {

cout << setw(4) << matrix[row][colum];//оператор выводит значение элемента матрицы в текущей позиции (matrix[row][colum]) с помощью cout. setw(4) используется для выравнивания значений, чтобы они занимали одинаковое количество символов, в данном случае 4 символа.

}

cout << endl;

}

}

void deleteMatrix(int\*\* matrix, int rows) {

for (int i = 0; i < rows; ++i) {//В этом цикле происходит освобождение памяти, выделенной под каждую строку матрицы.

delete[] matrix[i];

}

delete[] matrix;

}

int main() {

setlocale(LC\_ALL, "Rus");

int rows, colums;

cout << "Введите количество строк матрицы: ";

cin >> rows;

cout << "Введите количество столбцов матрицы: ";

cin >> colums;

int\*\* dynMatrix = new int\* [rows];//создает двумерный массив (матрицу) с помощью динамического выделения памяти.

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

dynMatrix[i] = new int[colums];

}

cout << "Введите элементы матрицы:" << endl;//заполняем массив

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

for (int colum = 0; colum < colums; ++colum) {

cout << "Элемент [" << row + 1 << "][" << colum + 1 << "]: ";

cin >> dynMatrix[row][colum];

}

}

Matrix\_1(dynMatrix, rows, colums);

cout << "Измененная матрица:" << endl;

Matrix\_2(dynMatrix, rows, colums);

deleteMatrix(dynMatrix, rows);

return 0;

}

