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»

Выполнил:

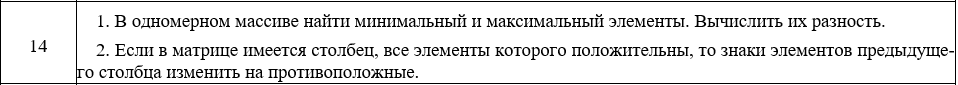
Студент 1 курса 10 группы

Сегренёв Кирилл Сергеевич

Преподаватель: асс. Андронова М.В.

2023, Минск

**ВАРИАНТ 14**



**Задание 1**

**Код:**

#include <iostream>

#include <ctime>

using namespace std;

// Прототипы

int searchMin(int\*, int);

int searchMax(int\*, int);

int searchSubtr(int&, int&);

int main() {

setlocale(LC\_ALL, "ru");

srand(time(0));

int n;

cout << "Введите длину последовательности: ";

cin >> n;

int\* arr = new int[n];

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

arr[i] = rand() % 51 - 10;

}

cout << "Последовательность: ";

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

cout << arr[i] << ' ';

}

cout << '\n';

//В данных функциях находим минимальное и максимальное число соответственно

int min = searchMin(arr, n);

int max = searchMax(arr, n);

// С помощью переменных, которые мы нашли, заходим в новую функцию, в которой считаем разницу между данными элементами

int subtr = searchSubtr(min, max);

cout << "Минимальный элемент последовательности: " << min << "\nМаксимальный элемент последовательности: " << max << "\nИх разность: " << subtr;

delete[] arr;

return 0;

}

int searchMin(int\* arr, int size)

{

int minNumber = arr[0];

for (int i = 0; i < size; i++) {// Поиск минимального числа последовательности

if (arr[i] < minNumber) {

minNumber = arr[i];

}

}

return minNumber;

}

int searchMax(int\* arr, int size)

{

int maxNumber = arr[0];

for (int i = 0; i < size; i++) {// Поиск максимального числа последовательности

if (arr[i] > maxNumber) {

maxNumber = arr[i];

}

}

return maxNumber;

}

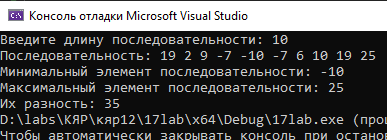
int searchSubtr(int& a, int& b)

{

return abs(a - b); // Возвращение модуля разности чисел

}

**Output:**



**Задание 2**

**Код:**

#include <iostream>

#include <ctime>

// Прототипы

void printMatrix(int\*\*, int&, int&);

int searchCol(int\*\*, int&, int&);

using namespace std;

int main() {

setlocale(LC\_ALL, "ru");

srand(time(0));

int n, m, index = 0, id = 0;

cout << "Введите количество строк и столбцов соответственно(через пробел): ";

cin >> m >> n;

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

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

arr[i] = new int[m];

}

for (int i = 0; i < n; i++) // Записываем в матрицу рандомные значения

{

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

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

}

cout << "\nНаальная матрица:\n";

printMatrix(arr, m, n); // С помощью функции printMatrix выводим начальную матрицу

id = searchCol(arr, n, m); // С помощью функции находим строку, в которой будут находится максимальные элементы

if(id != -1)

{

cout << "\nИзмененная матрица:\n";

if(id != 0)

{

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

arr[i][id - 1] \*= (-1);

}

}

else {

for (int i = 0; i < n; i++) { // Добавил условие для первого столбца. Когда положительные элементы находятся в первом столбце, изменяет 4 столбик.

arr[i][m-1] \*= (-1);

}

}

printMatrix(arr, m, n);

}

else { // Если в матрице не оказалось столбца с пол элементами, вывести на экран сообщение.

cout << "\nВ данной матрице нет столбца с положительными элементами." << endl;

}

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

{

delete[] arr[i];

}

delete[] arr;

return 0;

}

void printMatrix(int\*\* arr, int& sizeCol, int& sizeRow) {

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

for (int g = 0; g < sizeCol; g++) {

cout << arr[i][g] << '\t';

}

cout << '\n';

}

}

int searchCol(int\*\* arr, int& n, int& m)

{

int index = 0;

int id = -1;// Для конечного сравнения, чтобы вернуть правильное значение( номер столбца, -1 значит, что данного столбца нет )

for (int i = 0; i < m; i++) // Проверяем каждый элемент столбца на то, больше ли он 0 или нет

{

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

if (arr[j][i] > 0) {

index++;

}

if (index == n) {

id = i;

break;

}

}

if (index == m) {

break;

}

index = 0;

}

if (id != -1)

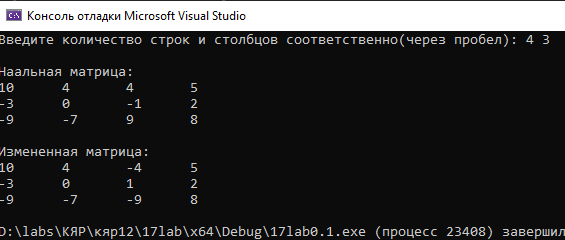
return id;

else

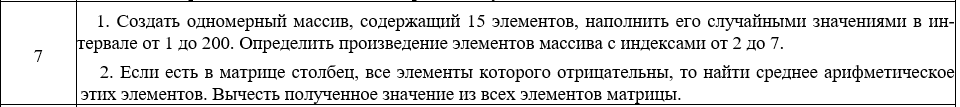
return -1;

}

**Output:**



**ДОПОЛНИТЕЛЬНЫЕ ЗАДАНИЯ**

**ВАРИАНТ 7** ****

**Задание 1**

**Код:**

#include <iostream>

#include<ctime>

using namespace std;

//7

double multi(int\*, int);

int main() {

setlocale(LC\_ALL, "ru");

srand(time(0));

const int n = 15;

int arr[n];

for (int i = 0; i < n; i++) { // Записываем рандомные значения в интервале от 1 до 200

arr[i] = rand() % 201 + 1;

}

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

cout << arr[i] << ' ';

}

cout << "\nПроизведение элементов с индексами от 2 до 7: " << multi(arr, n) << endl; // Выводим результат выполнения функции milti()

return 0;

}

double multi(int\* arr, int size) // Слишком большое значение, поэтому используем double

{

double multi = 1;

for (int i = 2; i <= 7; i++) { // Со второй позиции начинаем перемножать элементы

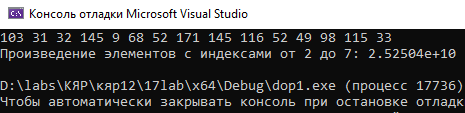
multi \*= arr[i];

}

return multi;

}

**Output:**

****

**Задание 2**

**Код:**

#include <iostream>

#include <ctime>

using namespace std;

void printMatrix(int\*\*, int&, int&);

void subtrElements(int\*\*, int&, int&, int&);

int searchCol(int\*\*, int&, int&);

int main() {

srand(time(0));

int n, m, index = 0, id = 0;

cin >> m >> n;

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

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

arr[i] = new int[m];

}

for (int i = 0; i < n; i++) // Записываем в матрицу рандомные значения

{

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

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

}

cout << "\nНачальная матрица:\n";

printMatrix(arr, n, m);

id = searchCol(arr, m, n); // С помощью данной функции находим идекс столбца, в котором все элементы будут отрицательны

if (id != -1) // Если его не будет, то вывести сообщение на экран

{

subtrElements(arr, n, m, id); // В данной функции мы находим среднее арифметическое найденого столбца и отнимаем от каждого элемента матрицы полученое число

cout << "\nМатрица после изменений:\n";

printMatrix(arr, n, m);

}

else {

cout << "\nВ данной матрице нет столбца с отрицательными элементами." << endl;

}

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

{

delete[] arr[i];

}

delete[] arr;

return 0;

}

void printMatrix(int\*\* arr, int& n, int& m) {

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

for (int g = 0; g < m; g++) {

cout << arr[i][g] << '\t';

}

cout << '\n';

}

}

int searchCol(int\*\* arr, int& m, int& n) {

int index = 0, id = -1;

for (int i = 0; i < m; i++) // Находим столбец, в котором все числа будут отрицательными

{

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

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

index++;

}

if (index == n) {

id = i;

break;

}

}

if (index == m) {

break;

}

index = 0;

}

if (id != -1)

return id;

else

return -1;

}

void subtrElements(int\*\* arr, int& n, int& m, int& id)

{

int average = 0;

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

average += arr[i][id];

}

average /= n; //Среднее арифметическое найденого столбца

for (int i = 0; i < n; i++) { // Отнимаем от каждого элемента матрицы среднее арифметическое

for (int g = 0; g < m; g++) {

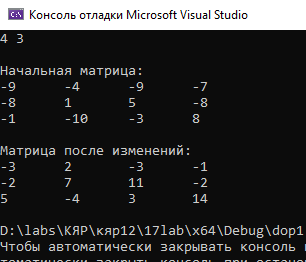
arr[i][g] -= average;

}

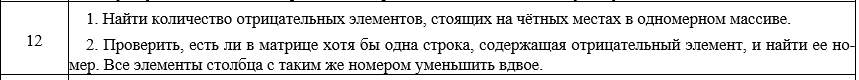
}

}

**Output:**

****

**ВАРИАНТ 12**

****

**Задание 1**

**Код:**

#include <iostream>

#include <ctime>

// 12

using namespace std;

//Прототип функции

int searchNegative(int\*, int&, int);

int main() {

setlocale(LC\_ALL, "ru");

srand(time(0));

int n, colMin = 0;

cout << "Введите длину последовательности: ";

cin >> n;

int\* arr = new int[n];

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

arr[i] = rand() % 51 - 25; // Ограничиваем подбор чисел для того, чтобы был больший шанс попасться отрицательному числу

}

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

cout << arr[i] << ' ';

colMin += searchNegative(arr, i, 0); // С помощью функции находим минимальный элемент и записываем значение индикатора в переменную colMin

}

cout << "\nКоличество отриц элементов, стоящих на четных местах: " << colMin << endl;

delete[] arr;

return 0;

}

int searchNegative(int\* arr, int& i, int minus) {

if ((i + 1) % 2 == 0) {

if (\*(arr+i) < 0) {

minus++; // Индикатор реагирует на то, появляется ли в последовательности отрицательное число

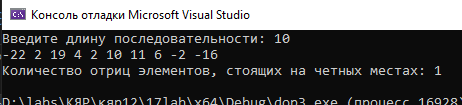
}

}

return minus; // Возвращаем данное значение

}

**Output:**

****

**Задание 2**

**Код:**

#include <iostream>

#include <ctime>

using namespace std;

void printMatrix(float\*\*, int&);

int searchStr(float\*\*, int&);

float division(float\*\*, int, int);

int main() {

setlocale(LC\_ALL, "ru");

srand(time(0));

int n, id = -1;

cout << "Введите размер квадратной матрицы: ";

cin >> n;

float\*\* arr = new float\* [n];

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

arr[i] = new float[n];

}

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

{

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

{

arr[i][j] = rand() % 31 - 5;

}

}

cout << "Начальная матрица: " << endl;

printMatrix(arr, n); // С помощью функции печатаем начальную матрицу на экран

id = searchStr(arr, n); // Переменной id(номер строки, которая содержит первый отрицательный элемент) присваиваем значение, которая вернет функция

if (id != -1) { // Проверяется условие, был ли в матрице отрицательный элемент

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

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

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

if (j == id) { // Если значение строки совпадает, зайти в функцию, которая вернет значение элемента матрица, уменьшенное в 2 рааз

cout << division(arr, i,j) << '\t';

}

else

cout << arr[i][j] << '\t';

}

cout << '\n';

}

cout << "\nНомер строки, в которой встречается первый отрицательный элемент: " << id + 1 << endl;

}

else

cout << "\nНет строк с отрицательными элементами((" << endl;

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

{

delete[] arr[i];

}

delete[] arr;

return 0;

}

void printMatrix(float\*\* arr, int& n) {

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

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

cout << arr[i][g] << '\t';

}

cout << '\n';

}

}

int searchStr(float\*\* arr, int& n) {

int id = -1, index = 0;

for (int i = 0; i < n; i++) // Цикл для поиска строки с отрицательным элементом

{

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

{

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

index++;

break; // Если он нашелся, то выйти из цикла

}

}

if (index == 1) {

id = i; // Присвоение переменной номера строки

break; // Выход из цикла

}

}

if (id != -1)

return id; // Вернуть значение строки

else

return -1;

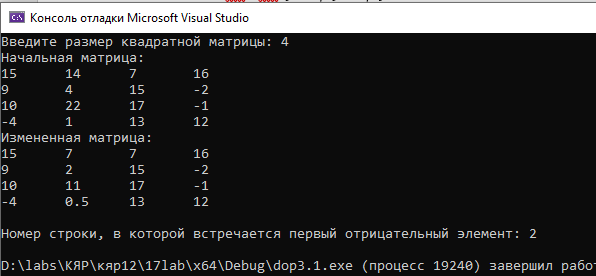
}

float division(float\*\* arr, int i, int j) {

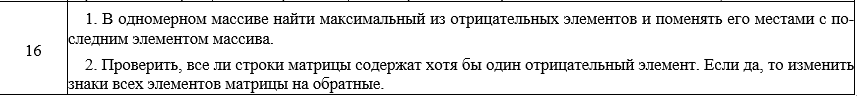
return \*(\*(arr + i) + j) / 2; // Уменьшение элемента матрицы в 2 раза

}

**Output:**



**ВАРИАНТ 16**



**Задание 1**

**Код:**

#include <iostream>

#include<ctime>

// 16

using namespace std;

int searchMaxNegative(int\*, int);

int main() {

setlocale(LC\_ALL, "ru");

srand(time(0));

int n;

cout << "Введите размер последовательности: ";

cin >> n;

int\* arr = new int[n];

cout << "Начальный массив: " << endl;

for (int i = 0; i < n; i++) { // Записываем в массив рандомные элементы

arr[i] = rand() % 51 - 25;

cout << arr[i] << ' ';

}

int indexMax = searchMaxNegative(arr, n); // С помощью функции находим индекс нужного элемента

cout << "\nЗначение максимального отриц элемента: " << \*(arr + indexMax) << "\nЕго индекс: " << indexMax + 1 << endl; // Вывод значения этого элемента и его индекс( +1 для более легко понимания) )

swap(arr[indexMax], arr[n - 1]); // Меняем местами последний и найденый элемент

cout << "Массив после изменений:\n";

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

cout << arr[i] << ' ';

}

cout << '\n';

delete[] arr;

return 0;

}

int searchMaxNegative(int\* arr, int n) {

int maxMin = -50; // Т.к мы ограничили свой рандом, то можем записать самое минимальное число, которое может выпасть

int indexMax;

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

if (arr[i] < 0) {

if (maxMin < arr[i]) { // Поиск максимального элемента среди отрицательных чисел

maxMin = arr[i];

indexMax = i;

}

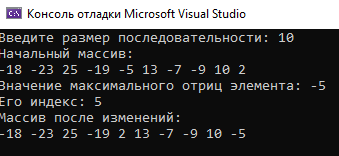
}

}

return indexMax;

}

**Output:**



**Задание 2**

**Код:**

#include <iostream>

#include <ctime>

using namespace std;

int searchNegative(int\*\*, int, int);

int multi(int\*\* arr, int i, int g);

int main() {

srand(time(0));

setlocale(LC\_ALL, "ru");

int n, m, id = 0;

cout << "Введите количество столбцов и количество строк через пробел: ";

cin >> m >> n;

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

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

arr[i] = new int[m];

}

cout << "Начальная матрица: " << endl;

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

{

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

{

arr[i][j] = rand() % 21 - 8; // Ограничения для rand, чтобы появились отрицательные числа

cout << arr[i][j] << '\t'; // Сразу же вывод на экран

}

cout << '\n';

}

cout << "\nИзмененная матрица: " << endl;

if (searchNegative(arr, n, m) == n) { // Сравниваем значение функции с количеством строк

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

for (int g = 0; g < m; g++) {

cout << multi(arr,i,g) << '\t'; // Заходим в функцию чтобы изменить элементы матрицы и сразу их выводим

}

cout << '\n';

}

}

else

cout << "В данной матрице не все строки с отрицательными элементами((";

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

{

delete[] arr[i];

}

delete[] arr;

return 0;

}

int searchNegative(int\*\* arr, int n, int m) // В данной функции находим, сколько строк у нас имеют хотябы 1 отриц элемент

{

int index = 0;

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

{

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

{

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

index++;

break;

}

}

}

return index;

}

int multi(int\*\* arr, int i, int g) {

return \*(\*(arr + i) + g) \* (-1); // Возвращаем элемент матрицы с противоположным знаком

}

**Output:**

