**Белорусский государственный технологический университет**

**Факультет информационных технологий**

**Кафедра ПИ**

Лабораторная работа № 17

По дисциплине «Основы алгоритмизации и программирования»

На тему «Указатели и ссылки при работе с функциями»

Выполнил:  
Студент 1 курса, 10 группа  
Короткевич Артём Сергеевич  
Преподаватель: асс. Харланович А.В

2024, Минск  
  
  
  
  
  
**Задание 1**  
Выполнив программу, записанную в правой части, изучить один из способов передачи *динамической* *матрицы* в функцию пользователя.  
Записать условие задачи.  
  
**Условие**  
Написать алгоритм, который будет высчитывать сумму всех элементов динамической матрицы   
**Код**

#include <iostream>  
using namespace std;

int sum(int\*\*, int, int);

int main()

{

int i, j;

int\*\* matr = new int\* [5];

for (i = 0; i < 5; i++)

matr[i] = new int[4];

for (i = 0; i < 5; i++)

{

for (j = 0; j < 4; j++)

{

matr[i][j] = i + j;

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

}

cout << endl;

}

cout << "S=" << sum(matr, 5, 4) << endl;

for (int i = 0; i < 5; i++)

delete matr[i];

delete[] matr;

}

int sum(int\*\* x, int n, int m)

{

int res = 0;

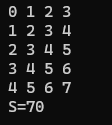
for (int i = 0; i < n; i++)

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

res += x[i][j];

return res;

}

**Результат**  
  
  
  
**Задание 2**Выполнив программу, записанную в правой части, изучить один из способов передачи *одномерного* *массива* в функцию, а также использование *указателя как результата* выполнения функции.  
  
**Код**

#include <iostream>  
int\* pfmin(int\* p, int n);

void main()

{

int masB[5] = { 4, 8, 2, 6, 4 };

(\*pfmin(masB, 5))++;

for (int i = 0; i < 5; i++)

std::cout << masB[i] << ' ';

}

int\* pfmin(int\* p, int n)

{

int\* pmin;

for (pmin = p; n > 0; p++, n--)

if (\*p < \*pmin)

pmin = p;

return pmin;

}

**Результат**

  
  
**Задание 3**  
Изучить использование *ссылки* как *результата* работы функции, выполнив программу, записанную в правой части.  
Опробовать работу программы с разными значениями массива **А**.  
  
**Код**

#include <iostream>  
using namespace std;

double& dmin(double A[], int size);

void main()

{

double s;

const int size = 5;

double A[] = { 5, 4.1, 3, 0.2, 11 };

s = dmin(A, size);

cout << s << endl;

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

cout << " " << A[i];

cout << endl;

dmin(A, size) = 1.0; // изменение минимума на значение 1.0

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

cout << " " << A[i];

}

double& dmin(double A[], int size)

{

int i, j = 0;

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

if (A[j] > A[i])

j = i;

return A[j];

}

**Результат**  
  
**Задание 4**  
Ознакомиться с использованием функций с результатом *логического* типа, опробовав работу программы в правой части.  
  
**Код**

#include <iostream>  
using namespace std;

bool is\_elem(int\* pA, int n, int iV);

void main()

{

setlocale(LC\_CTYPE, "Rus");

bool t; int k;

int A[] = { 5, 4, 3, 2, 11 };

cout << "Введите число ";

cin >> k;

t = is\_elem(A, sizeof(A) / sizeof(int), k);

if (t == true)

cout << "Число есть в массиве ";

else

cout << "Числа нет в массиве ";

}

bool is\_elem(int\* pA, int n, int iV)

{

bool bf = false;

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

if (pA[i] == iV)

{

bf = true;

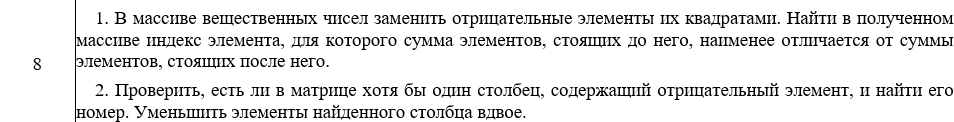
break;

}

return bf;

}

**Результат**  
  
  
  
**Задание 5**  
В соответствии со своим вариантом написать главную функцию, в которой имеются вызовы ***функций пользователя***, реализующих задачи из таблицы ниже. Ввод исходных данных и вывод результатов осуществить в главной функции, при этом использовать***динамические* *массивы***.  
Для передачи параметров в функции пользователя и возвращения результатов применить ***указатели*** и ***ссылки***.

**Вариант 8**  


**Задание 1**  
  
**Код**

#include <iostream>

using namespace std;

void processArray(double\* arr, int size, int& resultIndex) {

for (int i = 0; i < size; ++i) { // замена отрицательных элементов их квадратами

if (arr[i] < 0) {

arr[i] = arr[i] \* arr[i];

}

}

double totalSum = 0; // поиск индекса, где разница между суммами минимальна

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

totalSum += arr[i];

}

double leftSum = 0;

double minDifference = 1e9; // большое значение для инициализации

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

double rightSum = totalSum - leftSum - arr[i];

double difference = (leftSum > rightSum) ? (leftSum - rightSum) : (rightSum - leftSum); // модуль разности

if (difference < minDifference) {

minDifference = difference;

resultIndex = i;

}

leftSum += arr[i];

}

}

int main() {

setlocale(LC\_ALL, "Russian");

int size;

cout << "Введите размер массива: ";

cin >> size;

double\* arr = new double[size];

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

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

cin >> arr[i];

}

int resultIndex = 0;

processArray(arr, size, resultIndex);

cout << "Измененный массив: ";

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

cout << arr[i] << " ";

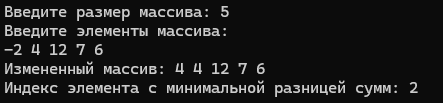
}

cout << endl;

cout << "Индекс элемента с минимальной разницей сумм: " << resultIndex << endl;

delete[] arr; // освобождаем динамическую память

return 0;  
}

**Результат**  
  
  
  
**Задание 2**  
  
**Код**

#include <iostream>

using namespace std;

void processMatrix(double\*\* matrix, int rows, int cols, int& Index) {

Index = -1;

for (int j = 0; j < cols; ++j) { // проверяем столбцы на наличие отрицательных элементов

bool hasNegative = false;

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

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

hasNegative = true;

break;

}

}

if (hasNegative) { // если найдётся столбец с отрицательным элементом, запоминаем его индекс и уменьшаем элементы вдвое

Index = j;

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

matrix[i][j] /= 2;

}

break; // будет достаточно найти только первый такой столбец

}

}

}

int main() {

setlocale(LC\_ALL, "Russian");

int rows, cols;

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

cin >> rows >> cols;

double\*\* matrix = new double\* [rows]; // создание динамической матрицы

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

matrix[i] = new double[cols];

}

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

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

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

cin >> matrix[i][j];

}

}

int Index = -1;

processMatrix(matrix, rows, cols, Index);

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

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

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

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

}

cout << endl;

}

if (Index != -1) {

cout << "Первый столбец с отрицательным элементом: " << Index << endl;

}

else {

cout << "Столбцов с отрицательными элементами нет" << endl;

}

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

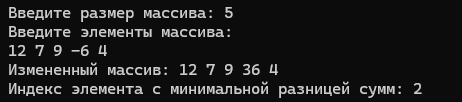
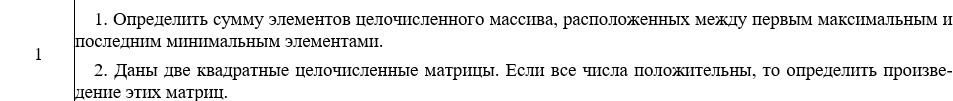
delete[] matrix[i];

}

delete[] matrix; // освобождаем динамическую память

return 0;

}

**Результат**  
  
  
  
**Дополнительные задания**  
  
**Вариант 1**  
**Задание 1**  
  
**Код**

#include <iostream>  
using namespace std;

int findSumBetween(int\* arr, int size) {

int maxIndex = -1, minIndex = -1;

int maxElement = arr[0], minElement = arr[0];

for (int i = 0; i < size; ++i) { // ищем первый максимальный и последний минимальный элементы

if (arr[i] > maxElement) {

maxElement = arr[i];

maxIndex = i;

}

if (arr[i] <= minElement) { // последний минимальный

minElement = arr[i];

minIndex = i;

}

}

if (maxIndex > minIndex) { // определяем порядок индексов

swap(maxIndex, minIndex);

}

int sum = 0;

for (int i = maxIndex + 1; i < minIndex; ++i) {

sum += arr[i];

}

return sum;

}

int main() {

setlocale(LC\_ALL, "Russian");

int size;

cout << "Введите размер массива: ";

cin >> size;

int\* arr = new int[size];

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

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

cin >> arr[i];

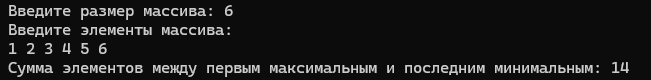
}

int sum = findSumBetween(arr, size);

cout << "Сумма элементов между первым максимальным и последним минимальным: " << sum << endl;

delete[] arr;

return 0;  
}

**Результат**  
  
  
  
**Задание 2**  
  
**Код**

#include <iostream>  
using namespace std;

bool AllPositive(int\*\* matrix, int size) {

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

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

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

return false;

}

}

}

return true;

}

void multiplyMatrices(int\*\* mat1, int\*\* mat2, int\*\* result, int size) {

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

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

result[i][j] = 0;

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

result[i][j] += mat1[i][k] \* mat2[k][j];

}

}

}

}

int main() {

setlocale(LC\_ALL, "Russian");

int size;

cout << "Введите размер матриц: ";

cin >> size;

int\*\* mat1 = new int\* [size]; // создаём матрицы

int\*\* mat2 = new int\* [size];

int\*\* result = new int\* [size];

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

mat1[i] = new int[size];

mat2[i] = new int[size];

result[i] = new int[size];

}

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

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

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

cin >> mat1[i][j];

}

}

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

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

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

cin >> mat2[i][j];

}

}

if (AllPositive(mat1, size) && AllPositive(mat2, size)) { // проверяем, что все числа положительные

multiplyMatrices(mat1, mat2, result, size);

cout << "Результат произведения матриц: " << endl;

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

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

cout << result[i][j] << " ";

}

cout << endl;

}

}

else {

cout << "Не все числа положительные, умножение невозможно" << endl;

}

// освобождаем динамичскую память матриц

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

delete[] mat1[i];

delete[] mat2[i];

delete[] result[i];

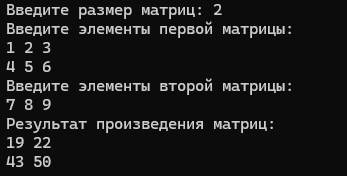
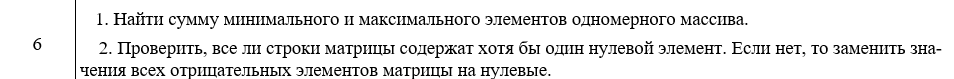
}

delete[] mat1;

delete[] mat2;

delete[] result;

return 0;  
}

**Результат**  
  
  
  
**Вариант 6**  
  
  
**Задание 1**  
  
**Код**

#include <iostream>  
using namespace std;

int massiv(int\* mas, int size) {

int min = mas[0], max = mas[0], sum = 0;

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

{

if (mas[i] < min) { // находим минимальный элемент

min = mas[i];

}

if (mas[i] > max) { // находим максимальный элемент

max = mas[i];

}

if (max > mas[i] || min < mas[i]) {

sum = max + min; // находим сумму

}

}

return sum;

}

int main()

{

setlocale(LC\_ALL, "Russian");

int size;

cout << "Введите длину массива: ";

cin >> size;

int\* mas = new int[size]; // создаём динамический массив

cout << "Введите элементы массива: ";

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

{

cin >> mas[i]; // вводим элементы массива

}

cout << "Сумма максимального и минимального элементов равна: " << massiv(mas, size);

delete[] mas; // очищаем память из массива

}

**Результат**  
  
  
  
**Задание 2**  
  
**Код**

#include <iostream>  
using namespace std;

void Zero(int\*\* matrix, int n, int m) {

bool zero = true; // осуществляем поиск с помощью флагов

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

bool zero1 = false;

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

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

zero1 = true;

break;

}

}

if (!zero1) {

zero = false;

break;

}

}

if (!zero) { // если элемент не равен 0, то меняем все отрицательные значения на нулевые

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

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

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

matrix[i][j] = 0;

}

}

}

}

}

int main()

{

setlocale(LC\_ALL, "Russian");

int n, m;

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

cin >> n >> m;

int\*\* matrix = new int\* [n]; // создаём динамический массив

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

matrix[i] = new int[m];

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

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

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

cin >> matrix[i][j];

}

}

Zero(matrix, n, m);

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

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

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

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

}

cout << endl;

}

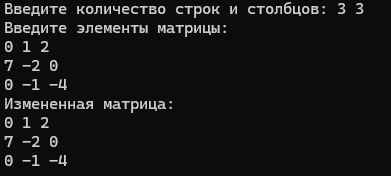
for (int i = 0; i < n; i++)

delete[] matrix[i];

delete[] matrix; // очищаем динамическую память

return 0;

}

**Результат**  
  
  
  
**Вариант 15**  
  
  
  
**Задание 1**  
  
 **Код**

#include <iostream>  
using namespace std;

int countNegativeAfterLastMin(int\* arr, int size) {

int minIndex = 0;

int minElement = arr[0];

for (int i = 0; i < size; ++i) { // найдём индекс последнего минимального элемента

if (arr[i] <= minElement) {

minElement = arr[i];

minIndex = i;

}

}

int negativeCount = 0;

for (int i = minIndex + 1; i < size; ++i) { // подсчитаем количество отрицательных элементов после последнего минимального

if (arr[i] < 0) {

++negativeCount;

}

}

return negativeCount;

}

int main() {

setlocale(LC\_ALL, "Russian");

int size;

cout << "Введите размер массива: ";

cin >> size;

int\* arr = new int[size];

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

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

cin >> arr[i];

}

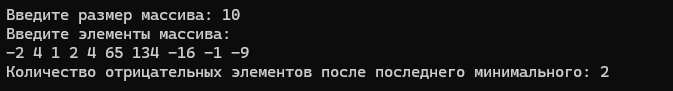
int negativeCount = countNegativeAfterLastMin(arr, size);

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

delete[] arr;

return 0;

}

**Результат**  
  
  
  
**Задание 2**  
  
 **Код**

#include <iostream>  
using namespace std;  
bool hasPositiveElement(int\*\* matrix, int rows, int col) { // проверяем, содержит ли столбец хотя бы один положительный элемент

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

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

return true;

}

}

return false;

}

void replaceNegativesWithAbs(int\*\* matrix, int rows, int col) { // заменяем отрицательные элементы в столбце на их модули

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

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

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

}

}

}

int main() {

setlocale(LC\_ALL, "Russian");

int rows, cols;

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

cin >> rows >> cols;

int\*\* matrix = new int\* [rows]; // создаём динамическую матрицу

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

matrix[i] = new int[cols];

}

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

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

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

cin >> matrix[i][j];

}

}

bool ColumnsHavePositive = true; // Проверка каждого столбца

int firstInvalidCol = -1;

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

if (!hasPositiveElement(matrix, rows, j)) {

ColumnsHavePositive = false;

firstInvalidCol = j;

break;

}

}

if (ColumnsHavePositive) {

cout << "Все столбцы содержат хотя бы один положительный элемент." << endl;

}

else {

cout << "Первый столбец без положительных элементов: " << firstInvalidCol << endl;

replaceNegativesWithAbs(matrix, rows, firstInvalidCol);

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

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

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

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

}

cout << endl;

}

}

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

delete[] matrix[i];

}

delete[] matrix;

return 0;

}

**Результат**  
  
