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 7 группы

Ленкевич Павел Андреевич

Преподаватель: асс. Андронова М.В.

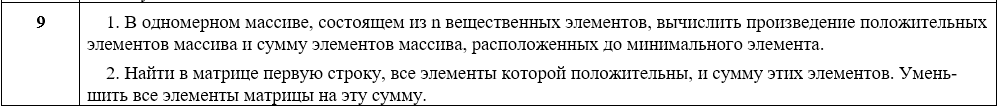
2023, Минск

**Задание**

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++**.

**Вариант 9.**

**Условие:**



**Задание 1.**

#include <iostream> // Подключение библиотек

using namespace std; // Позволяет не писать std перед операторами ввода-вывода

int main() {

setlocale(LC\_CTYPE, "Russian"); // Подключает русский язык

int k;

cout << "Введите размер массива: "; cin >> k;

int\* arr = (int\*)malloc(k \* sizeof(int)); // Выделение памяти под массив

cout << "Введите элементы массива: "; // Ввод элементов массива

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

cin >> arr[i];

}

int positive\_count = 0, sum = 0, last\_zero\_index = -1;

for (int i = 0; i < k; i++) { // Подсчет положительных элементов и индекса последнего нулевого элемента

if (arr[i] > 0) {

positive\_count++;

}

if (arr[i] == 0) {

last\_zero\_index = i;

}

}

for (int i = last\_zero\_index + 1; i < k; i++) { // Подсчет суммы элементов после последнего нулевого элемента

sum += arr[i];

}

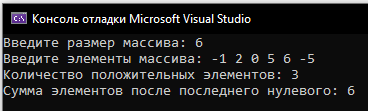
cout << "Количество положительных элементов: " << positive\_count << endl;

cout << "Сумма элементов после последнего нулевого: " << sum << endl;

free(arr); // Освобождение памяти

return 0;

}



**Задание 2.**

#include <iostream> } // Подключаем библиотеки

using namespace std; // Позволяет не писать std перед операторами ввода-вывода

int main() {

setlocale(LC\_CTYPE, "Russian"); // Подключает русский язык

int rows, cols;

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

cin >> rows >> cols;

int\*\* matrix = new int\* [rows]; // Создание динамической матрицы

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

matrix[i] = new int[cols];

cout << "Введите элементы матрицы:\n"; // Заполнение матрицы

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

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

cin >> matrix[i][j];

int sum = 0;

for (int i = 0; i < rows; ++i) { // Поиск первой строки, где все элементы положительны

bool allPositive = true;

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

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

allPositive = false;

break;

}

sum += matrix[i][j];

}

if (allPositive) break;

}

for (int i = 0; i < rows; ++i) // Уменьшение всех элементов матрицы на найденную сумму

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

matrix[i][j] -= sum;

cout << "Обновленная матрица:\n"; // Вывод обновленной матрицы

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

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

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

cout << '\n';

}

for (int i = 0; i < rows; ++i) // Освобождение памяти

delete[] matrix[i];

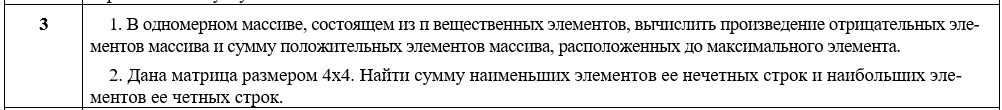
delete[] matrix;

return 0;

}

**Дополнительные варианты**

**Вариант 3**

****

**Задание 1.**

#include <iostream> // Подключение библиотек

#include <cstdlib>

using namespace std; // Позволяет не писать std перед операторами ввода-вывода

int main() {

setlocale(LC\_CTYPE, "Russian"); // Подключает русский язык

int p;

cout << "Введите размер массива: ";

cin >> p;

double\* arr = (double\*)malloc(p \* sizeof(double)); // Выделение памяти под массив

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

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

cin >> arr[i];

}

double prod = 1.0, sum = 0.0;

double max\_val = arr[0];

int max\_index = 0;

for (int i = 0; i < p; i++) { // Поиск максимального элемента и вычисление произведения отрицательных элементов

if (arr[i] > max\_val) {

max\_val = arr[i];

max\_index = i;

}

if (arr[i] < 0) {

prod \*= arr[i];

}

}

for (int i = 0; i < max\_index; i++) { // Вычисление суммы положительных элементов до максимального

if (arr[i] > 0) {

sum += arr[i];

}

}

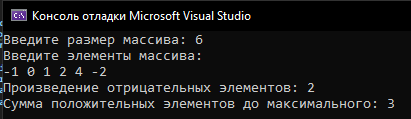
cout << "Произведение отрицательных элементов: " << prod << "\n"; // Вывод результатов

cout << "Сумма положительных элементов до максимального: " << sum << "\n";

free(arr); // Освобождение памяти

return 0;

}

****

**Задание 2.**

#include <iostream>

using namespace std;

int main() {

setlocale(LC\_CTYPE, "Russian"); // Подключает русский язык

int n = 4, m = 4; // Задаем размерность матрицы 4x4

int\*\* arr = new int\* [n]; // Создаем динамический двумерный массив

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

arr[i] = new int[m];

cout << "Введите элементы матрицы 4x4:\n";

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

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

cin >> arr[i][j];

int sum = 0; // Переменная для хранения суммы экстремальных элементов

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

int ext = arr[i][0]; // Начальное значение для поиска экстремального элемента

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

if (i % 2 == 0 && arr[i][j] > ext) // Для четных строк ищем максимальный элемент

ext = arr[i][j];

else if (i % 2 != 0 && arr[i][j] < ext) // Для нечетных строк ищем минимальный элемент

ext = arr[i][j];

}

sum += ext; // Добавляем экстремальный элемент к сумме

}

cout << "Сумма наименьших элементов нечетных строк и наибольших элементов четных строк равна " << sum << endl; // Выводим сумму экстремальных элементов

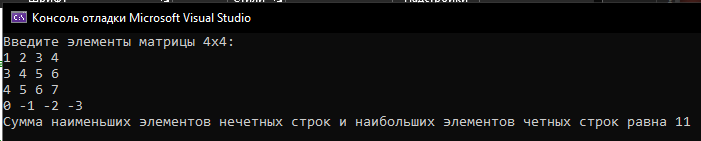
for (int i = 0; i < n; i++) // Освобождаем память, выделенную под массив

delete[] arr[i];

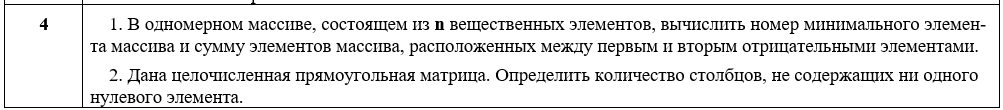
delete[] arr;

return 0;

}

****

**Вариант 4.**

****

**Задание 1.**

#include <iostream> // Подключение библиотек

#include <cstdlib>

using namespace std; // Позволяет не писать std перед операторами ввода-вывода

int main() {

setlocale(LC\_CTYPE, "Russian"); // Подключает русский язык

int n;

cout << "Введите размер массива: ";

cin >> n;

float\* arr = (float\*)malloc(n \* sizeof(float)); // Выделение памяти под массив

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

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

cin >> arr[i];

}

int minIndex = 0;

for (int i = 1; i < n; i++) { // Поиск минимального элемента

if (arr[i] < arr[minIndex]) {

minIndex = i;

}

}

cout << "Индекс минимального элемента: " << minIndex << "\n";

int firstNeg = -1, secondNeg = -1;

for (int i = 0; i < n; i++) { // Поиск первого и второго отрицательных элементов

if (arr[i] < 0) {

if (firstNeg == -1) {

firstNeg = i;

}

else {

secondNeg = i;

break;

}

}

}

if (firstNeg != -1 && secondNeg != -1) { // Вычисление суммы элементов между первым и вторым отрицательными элементами

float sum = 0;

for (int i = firstNeg + 1; i < secondNeg; i++) {

sum += arr[i];

}

cout << "Сумма элементов между первым и вторым отрицательными элементами: " << sum << "\n";

}

else {

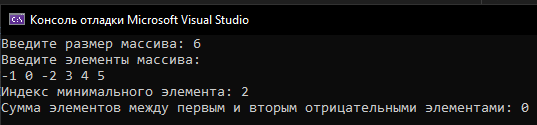
cout << "В массиве нет двух отрицательных элементов.\n";

}

free(arr); // Освобождение памяти

return 0;

}

****

**Задание 2.**

#include <iostream> // Подключение библиотек

using namespace std; // Позволяет не писать std перед операторами ввода-вывода

int main() {

setlocale(LC\_CTYPE, "Russian"); // Подключает русский язык

int rows, cols;

cout << "Введите количество строк: ";

cin >> rows;

cout << "Введите количество столбцов: ";

cin >> cols;

int\*\* matrix = new int\* [rows]; // Создание динамического двумерного массива

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

matrix[i] = new int[cols];

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

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

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

cin >> matrix[i][j];

int count = 0;

for (int j = 0; j < cols; ++j) { // Проверка каждого столбца на наличие нулевых элементов

bool hasZero = false;

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

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

hasZero = true;

break;

}

}

if (!hasZero) ++count; // Если в столбце нет нулевых элементов, увеличиваем счетчик

}

cout << "Количество столбцов, не содержащих ни одного нулевого элемента: " << count << endl; // Вывод результата

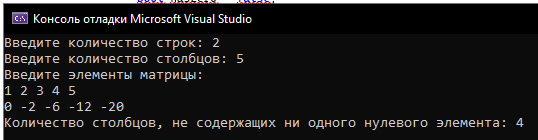
for (int i = 0; i < rows; ++i) // Освобождение памяти

delete[] matrix[i];

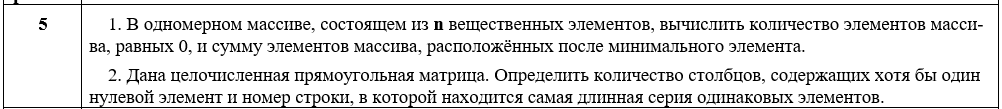
delete[] matrix;

return 0;

}

****

**Вариант 5.**

****

**Задание 1.**

#include <iostream> // Подключение библиотек

#include <cstdlib>

using namespace std; // Позволяет не писать std перед операторами ввода-вывода

int main() {

setlocale(LC\_CTYPE, "Russian"); // Подключает русский язык

int n;

cout << "Введите размер массива: ";

cin >> n;

float\* arr = (float\*)malloc(n \* sizeof(float)); // Выделение памяти под массив

for (int i = 0; i < n; i++) { // Ввод элементов массива

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

cin >> arr[i];

}

float min\_val = arr[0]; // Поиск минимального элемента и его индекса

int min\_index = 0;

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

if (arr[i] < min\_val) {

min\_val = arr[i];

min\_index = i;

}

}

int zero\_count = 0; // Подсчет количества нулевых элементов и суммы элементов после минимального

float sum = 0;

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

if (arr[i] == 0) {

zero\_count++;

}

if (i > min\_index) {

sum += arr[i];

}

}

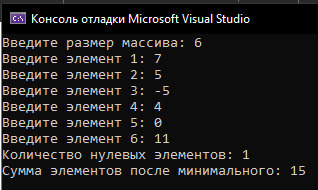
cout << "Количество нулевых элементов: " << zero\_count << endl;

cout << "Сумма элементов после минимального: " << sum << endl;

free(arr); // Освобождение памяти

return 0;

}

****

**Задание 2.**

#include <iostream> // Подключение библиотек

using namespace std; // Позволяет не писать std перед операторами ввода-вывода

int main() {

setlocale(LC\_CTYPE, "Russian"); // Подключает русский язык

int rows, cols;

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

cin >> rows >> cols;

int\*\* matrix = new int\* [rows]; // Создание динамического массива

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

matrix[i] = new int[cols];

cout << "Введите элементы матрицы:\n"; // Заполнение матрицы

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

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

cin >> matrix[i][j];

int zero\_cols = 0; // Подсчет столбцов с нулевыми элементами

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

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

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

zero\_cols++;

break;

}

}

}

int max\_series\_row = 0, max\_series\_length = 0; // Поиск строки с самой длинной серией одинаковых элементов

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

int current\_length = 1;

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

if (matrix[i][j] == matrix[i][j - 1])

current\_length++;

else

current\_length = 1;

if (current\_length > max\_series\_length) {

max\_series\_length = current\_length;

max\_series\_row = i;

}

}

}

cout << "Количество столбцов, содержащих хотя бы один нулевой элемент: " << zero\_cols << "\n";

cout << "Номер строки, в которой находится самая длинная серия одинаковых элементов: " << max\_series\_row + 1 << "\n";

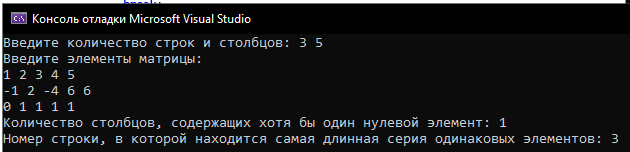
for (int i = 0; i < rows; ++i) // Освобождение памяти

delete[] matrix[i];

delete[] matrix;

return 0;

}

****