Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра программной инженерии

 Лабораторная работа 15

По дисциплине «Основы алгоритмизации и программирования»

На тему «Динамическое выделение памяти»

Выполнила:

Студентка 1 курса 10 группы

Рублевская Маргарита Владимировна

Преподаватель: асс. Андронова М.В.

2023, Минск

**Вариант№ 13**

1. 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++.

1. В одномерном массиве, состоящем из вещественных элементов, вычислить сумму элементов массива с нечетными номерами и сумму элементов массива, расположенных между первым и последним отрицательными элементами.

Код:

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

#include<iomanip> // Подключаем библиотеку для работы с setw()

// Подключаем заголовочные файлы для работы с функциями для работы со случайными числами и временем.

#include<cstdlib>

#include<ctime>

int main()

{

srand(time(0)); // Инициализация генератора случайных чисел

using namespace std;

setlocale(LC\_ALL, "Russian");

// Объявляем переменные

double\* A, sum\_odd = 0, sum\_negative = 0;

int n, first\_negative = -1, last\_negative = -1;

cout << "Введите размер массива: ";

cin >> n;

A = (double\*)malloc(sizeof(double) \* n); // Выделение динамической памяти для массива

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

A[i] = (((rand() - (RAND\_MAX / 2)) % 100)) / 10.0; // Используем числа с плавающей запятой для элементов массива

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

}

// Вычисляем сумму элементов с нечетными номерами

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

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

sum\_odd += A[i];

}

}

// Вычисляем сумму элементов между первым и последним отрицательными элементами

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

if (A[i] < 0) {

if (first\_negative == -1) {

first\_negative = i;

}

last\_negative = i;

}

}

if (first\_negative != -1 && last\_negative != -1) {

for (int i = first\_negative + 1; i < last\_negative; i++) {

sum\_negative += A[i];

}

}

cout << "\nСумма элементов с нечетными номерами: " << sum\_odd << endl;

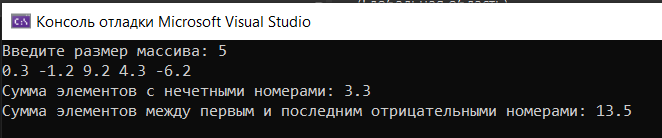
cout << "Сумма элементов между первым и последним отрицательными номерами: " << sum\_negative << endl;

free(A); // Освобождение динамической памяти, которую мы использовали для массива

return 0;

}

Консоль VS:



2. Найти в матрице первую строку, все элементы которой равны нулю. Все элементы столбца с таким же номером уменьшить вдвое.

Код:

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

#include <iomanip> // Подключаем библиотеку для работы с setw()

// Подключаем заголовочные файлы для работы с функциями для работы со случайными числами и временем.

#include <cstdlib>

#include <ctime>

int main() {

srand(time(0)); // Инициализация генератора случайных чисел

setlocale(LC\_ALL, "Russian");

using namespace std;

// Объявляем переменные

int n, m;

bool condition = false; // Объявляем переменную с именем condition типа bool со значением false

cout << "Количество строк: ";

cin >> n;

cout << "Количество столбцов: ";

cin >> m;

// Создаем динамическую матрицу размера n x m

int\*\* A = new int\* [n]; // Выделение памяти под указатели на массивы

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

A[i] = new int[m]; // Выделение памяти под каждый массив в матрице

}

// Выводим матрицу на экран после ее заполнения случайными числами

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

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

A[i][j] = rand() % 3;

cout << setw(4) << A[i][j] << " "; }

cout << endl;

}

// Поиск первой строки, где все элементы равны нулю

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

condition = true;

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

if (A[i][j] != 0) {

condition = false;

break;

}

}

if (condition) {

// Уменьшение элементов столбца вдвое

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

A[k][i] /= 2;

}

break;

}

}

// Выводим матрицу на экран после ее преобразования

if (condition) {

cout << "\n\nОбновленная матрица: \n";

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

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

cout << setw(4) << A[i][j] << " "; }

cout << endl;

}

}

// Освобождение динамической памяти

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

delete[] A[i];

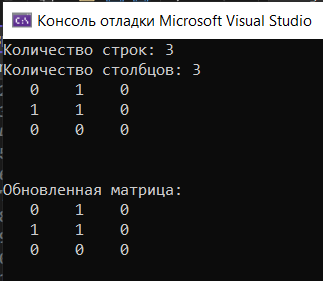
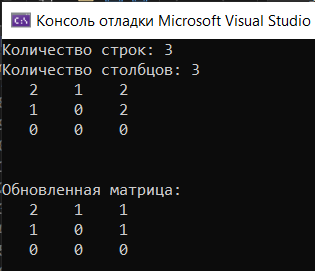
}

delete[] A;

return 0;

}

Консоль VS:

**Вариант№ 9**

1. В одномерном массиве, состоящем из n вещественных элементов, вычислить произведение положительных элементов массива и сумму элементов массива, расположенных до минимального элемента.

Код:

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

// Подключаем заголовочные файлы для работы с функциями для работы со случайными числами и временем.

#include <cstdlib>

#include <ctime>

#include <climits> // Для использования констант-ограничений

int main() {

srand(time(0)); // Инициализация генератора случайных чисел

setlocale(LC\_ALL, "Russian");

using namespace std;

// Объявляем переменные

double\* A;

int n;

cout << "Введите размер массива: ";

cin >> n;

A = (double\*)malloc(sizeof(double) \* n); // Выделение динамической памяти

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

A[i] = static\_cast<double>(rand() % 100 - 50) / 10.0; // Генерация вещественных значений от -5.0 до 5.0

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

}

// Вычисляем произведение положительных элементов

double positiveProduct = 1.0;

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

if (A[i] > 0) {

positiveProduct \*= A[i];

}

}

// Находим минимальный элемента и суммы элементов до него

double sum\_before\_min = 0;

int min\_index = 0;

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

{

if (A[i] < A[min\_index])

{

min\_index = i;

}

}

for (int i = 0; i < min\_index; i++)

{

sum\_before\_min += A[i];

}

cout << "\nПроизведение положительных элементов: " << positiveProduct << endl;

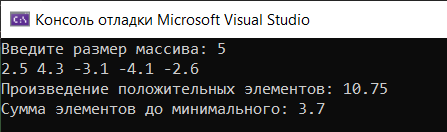
cout << "Сумма элементов до минимального: " << sum\_before\_min << endl;

free(A); // Освобождение динамической памяти

return 0;

}

Консоль VS:



2. Найти в матрице первую строку, все элементы которой положительны, и сумму этих элементов. Уменьшить все элементы матрицы на эту сумму.

Код:

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

#include <iomanip> // Подключаем библиотеку для работы с setw()

// Подключаем заголовочные файлы для работы с функциями для работы со случайными числами и временем.

#include <cstdlib>

#include <ctime>

int main() {

srand(time(0)); // Инициализация генератора случайных чисел

setlocale(LC\_ALL, "Russian");

using namespace std;

// Объявляем переменные

int n, m;

bool condition = false; // Объявляем переменную с именем condition типа bool со значением false

cout << "Количество строк: ";

cin >> n;

cout << "Количество столбцов: ";

cin >> m;

// Создаем динамическую матрицу размера n x m

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

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

A[i] = new int[m];

}

// Выводим матрицу на экран после ее заполнения случайными числами

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

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

A[i][j] = rand() % 10;

cout << setw(4) << A[i][j] << " ";

}

cout << endl;

}

int positiveSum = 0;

bool found = false;

// Поиск первой строки, где все элементы положительные и вычисление их суммы

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

condition = true;

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

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

condition = false;

break;

}

positiveSum += A[i][j];

}

if (condition) {

found = true;

// Уменьшение всех элементов матрицы на сумму положительных элементов

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

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

A[x][y] -= positiveSum;

}

}

break;

}

}

// Выводим матрицу на экран после ее преобразования

if (found) {

cout << "\n\nОбновленная матрица: \n";

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

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

cout << setw(4) << A[i][j] << " ";

}

cout << endl;

}

}

// Освобождение динамической памяти

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

delete[] A[i];

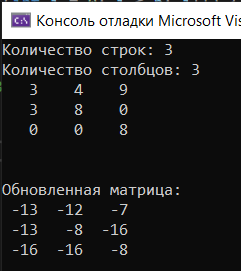
}

delete[] A;

return 0;

}

Консоль VS:



**Вариант№ 6**

1. В одномерном массиве, состоящем из n вещественных элементов, вычислить минимальный элемент массива и сумму элементов, расположенных между первым и последним положительными элементами.

Код:

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

// Подключаем заголовочные файлы для работы с функциями для работы со случайными числами и временем.

#include <cstdlib>

#include <ctime>

int main() {

srand(time(0)); // Инициализация генератора случайных чисел

setlocale(LC\_ALL, "Russian");

using namespace std;

// Объявляем переменные

double\* A;

int n;

cout << "Введите размер массива: ";

cin >> n;

A = new double[n]; // Выделение динамической памяти

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

A[i] = static\_cast<double>(rand() % 100 - 50) / 10.0; // Генерация вещественных значений от -5.0 до 5.0

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

}

// Вычисление минимального элемента

double min = A[0]; // Предполагаем, что первый элемент массива является минимальным

for (int i = 1; i < n; i++) { // Начинаем цикл с элемента с индексом 1, так как первый уже рассмотрен

if (A[i] < min) {

min = A[i];

}

}

// Вычисляем сумму элементов между первым и последним положительными элементами

double sumBetweenPositives = 0.0;

int firstPositiveIndex = -1, lastPositiveIndex = -1;

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

if (A[i] > 0) {

if (firstPositiveIndex == -1) {

firstPositiveIndex = i;

}

lastPositiveIndex = i;

}

}

if (firstPositiveIndex != -1 && lastPositiveIndex != -1) {

for (int i = firstPositiveIndex + 1; i < lastPositiveIndex; i++) {

sumBetweenPositives += A[i];

}

}

cout << "\nМинимальный элемент: " << min << endl;

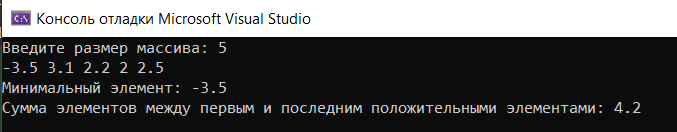
cout << "Сумма элементов между первым и последним положительными элементами: " << sumBetweenPositives << endl;

free(A); // Освобождение динамической памяти

return 0;

}

Консоль VS:



2. Проверить, есть ли в матрице хотя бы одна строка, содержащая отрицательный элемент, и найти ее номер. Все элементы столбца с таким же номером уменьшить вдвое.

Код:

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

#include <iomanip> // Подключаем библиотеку для работы с setw()

// Подключаем заголовочные файлы для работы с функциями для работы со случайными числами и временем.

#include <cstdlib>

#include <ctime> using namespace std;

int main() {

srand(time(0)); // Инициализация генератора случайных чисел

setlocale(LC\_ALL, "Russian");

// Объявляем переменные

int n, m, i;

int columnToReduce = -1;

bool found = false; // Объявляем переменную с именем found типа bool со значением false

cout << "Количество строк: ";

cin >> n;

cout << "Количество столбцов: ";

cin >> m;

// Создание динамической матрицы размера n x m

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

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

A[i] = new int[m];

}

// Выводим матрицу на экран после ее заполнения случайными числами

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

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

A[i][j] = rand() % 5 - 2; // Заполнение случайными значениями от -2 до 2

cout << setw(4) << A[i][j] << " ";

}

cout << endl;

}

// Поиск строки с отрицательным элементом

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

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

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

found = true;

columnToReduce = j; // Сохранение номера столбца для уменьшения

break;

}

}

if (found) {

break;

}

}

// Уменьшение всех элементов столбца вдвое, если найдена строка с отрицательным элементом

if (found) {

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

A[i][columnToReduce] /= 2;

}

// Выводим матрицу на экран после ее преобразования

cout << "\nОбновленная матрица: \n";

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

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

cout << setw(4) << A[i][j] << " ";

}

cout << endl;

}

}

else {

cout << "\n\nОтрицательных элементов не найдено." << endl;

}

// Освобождение динамической памяти

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

delete[] A[i];

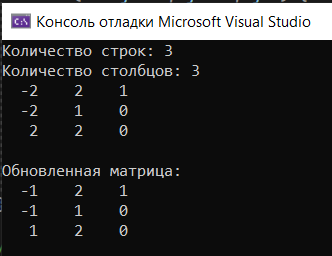
}

delete[] A;

return 0;

}

Консоль VS:



**Вариант№ 10**

1. В одномерном массиве, состоящем из n вещественных элементов, вычислить сумму положительных элементов массива и произведение элементов массива, расположенных между максимальным по модулю и минимальным по модулю элементами.

Код:

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

// Подключаем заголовочные файлы для работы с функциями для работы со случайными числами и временем.

#include <cstdlib>

#include <ctime>

int main() {

srand(time(0)); // Инициализация генератора случайных чисел

setlocale(LC\_ALL, "Russian");

using namespace std;

// Объявляем переменные

double\* A;

int n;

cout << "Введите размер массива: ";

cin >> n;

A = new double[n]; // Выделение динамической памяти

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

A[i] = (rand() % 100 - 50) / 10.0; // Генерация вещественных значений от -5.0 до 5.0

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

}

// Вычисляем сумму положительных элементов

double positiveSum = 0.0;

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

if (A[i] > 0) {

positiveSum += A[i];

}

}

// Вычисляем произведение элементов между максимальным и минимальным по модулю

int maxIndex = 0, minIndex = 0;

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

{

if (abs(A[i]) > abs(A[maxIndex])) // Если модуль текущего элемента больше модуля максимального

{

maxIndex = i; // Запоминаем индекс текущего элемента как индекс максимального

}

if (abs(A[i]) < abs(A[minIndex])) // Если модуль текущего элемента меньше модуля минимального

{

minIndex = i; // Запоминаем индекс текущего элемента как индекс минимального

}

}

double productBetweenExtremes = 1;

if (maxIndex < minIndex) // Если максимальный элемент находится левее минимального

{

for (int i = maxIndex + 1; i < minIndex; i++) // Проходим по элементам между ними

{

productBetweenExtremes \*= A[i]; // пПремножаем их

}

}

else // Если минимальный элемент находится левее максимального

{

for (int i = minIndex + 1; i < maxIndex; i++) // Проходим по элементам между ними

{

productBetweenExtremes \*= A[i]; // Перемножаем их

}

}

cout << "\nСумма положительных элементов: " << positiveSum << endl;

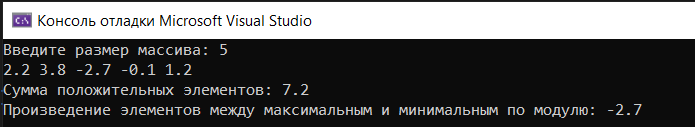
cout << "Произведение элементов между максимальным и минимальным по модулю: " << productBetweenExtremes << endl;

delete[] A; // Освобождение динамической памяти

return 0;

}

Консоль VS:



2. Найти в матрице первую строку, все элементы которой отрицательны. Увеличить все элементы матрицы на значение первого элемента найденной строки.

Код:

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

#include <iomanip> // Подключаем библиотеку для работы с setw()

#include <ctime> // Подключаем библиотеку для работы с случайными числами.

int main() {

srand(time(0)); // Инициализация генератора случайных чисел

setlocale(LC\_ALL, "Russian");

using namespace std;

// Объявляем переменные

int n, m;

bool condition = false; // Объявляем переменную с именем condition типа bool со значением false

cout << "Количество строк: ";

cin >> n;

cout << "Количество столбцов: ";

cin >> m;

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

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

A[i] = new int[m];

}

// Выводим матрицу на экран после ее заполнения случайными числами

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

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

A[i][j] = rand() % 10 - 5;

cout << setw(4) << A[i][j] << " ";

}

cout << endl;

}

// Находим первую строку, все элементы которой отрицательны

int firstNegativeRow = -1; // Индекс первой строки с отрицательными значениями

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

condition = true; // Предполагаем, что все элементы отрицательны

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

if (A[i][j] >= 0) {

condition = false; // Если хотя бы один элемент неотрицательный, то строка не удовлетворяет условию

break;

}

}

if (condition) {

firstNegativeRow = i; // Запоминаем индекс первой строки с отрицательными значениями и выходим из цикла

break;

}

}

if (firstNegativeRow != -1) {

cout << "Первая строка со всеми отрицательными элементами: ";

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

cout << A[firstNegativeRow][j] << " ";

}

cout << endl;

// Увеличиваем все элементы матрицы на значение первого элемента найденной строки

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

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

A[i][j] += A[firstNegativeRow][0];

}

}

// Выводим матрицу на экран после ее преобразования

cout << "Матрица после увеличения на значение первого элемента найденной строки: " << endl;

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

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

cout << setw(4) << A[i][j] << " ";

}

cout << endl;

}

}

else {

cout << "В матрице нет строк, удовлетворяющих условию." << endl;

}

// Освобождение динамической памяти

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

delete[] A[i];

}

delete[] A;

return 0;

}

Консоль VS:

