**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:

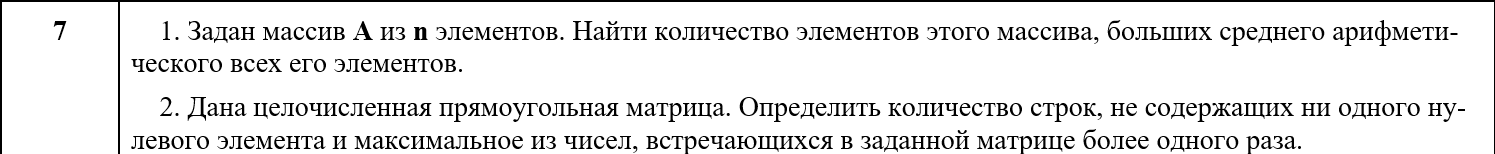
Студент 1 курса 10 группы

Мамонько Денис Александрович

Преподаватель: асс. Андронова М.В.

2023, Минск

**Вариант 7**

****

**Задание 1**

**Код программы:**

#include <iostream>//подключение библиотеки ввода/вывода

using namespace std;//чтобы не писать каждый раз std

int main()//объявляем главную функцию

{

setlocale(LC\_ALL, "RU");//подключаем кириллицу

int\* ptr, n;//объявляем переменные

//вводим размер массива

cout << "Введите размер массива: ";

cin >> n;

if (!(ptr = (int\*)malloc(n \* sizeof(int)))) //выделение памяти и проверка,

{ //достаточно ли для нее места

cout<<"Недостаточно памяти";

return 0;

}

//вводим элементы массива

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

{

cout << "Введите элементы массива " << i + 1 << " : ";

cin >> \*(ptr + i);

}

//иниициализируем переменную суммы

int sum = 0;

//выводим введенный массива

cout << "Введенный массив: ";

//находим сумму элементов массива

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

{

cout << \* (ptr + i) << ' ';

sum += \*(ptr + i);

}

//инициализируем переменную среднего арифметического и счетчик

double number = sum / n;

int count = 0;

//с помощью цикла определяем больше или меньше элемент среднего арифметического

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

{

if (\*(ptr + i) > number) {

count++;

}

else

continue;

}

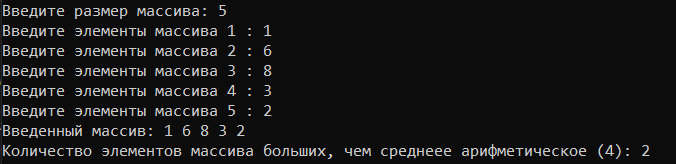
cout << endl;

cout << "Количество элементов массива больших, чем среднеее арифметическое ("<<number<<"): " << count;//выводим число элементов

free(ptr);//освобождение динамической памяти

}

**Результат программы:**

****

**Задание 2**

**Код программы:**

#include <iostream>//подключение библиотеки ввода/вывода

using namespace std;//чтобы не писать каждый раз std

int main()//объявляем главную функцию

{

setlocale(LC\_ALL, "Russian");//подключаем кириллицу

//ининициализируем и вводим данные матрицы

int n, m;

cout << "Введите количество строк матрицы: ";

cin >> n;

cout << "Введите количество столбцов матрицы: ";

cin >> m;

int\*\* A = new int\* [n];//выделение динамической памяти

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

{

A[i] = new int[n];

}

cout << "Введите матрицу: "<<endl;//вводим элементы матрицы

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

{

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

{

cout<<"A("<<i+1<<", "<<j+1<<"): ";

cin >> A[i][j];

}

}

cout << "Исходная матрица: "<<endl;//выводим исходную матрицу на экран

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

{

cout << "[";

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

{

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

}

cout << "]" << endl;

}

//инициализируем переменную, которая будет искать строку, не содержащую 0

int str = 0;

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

{

int a = 0;

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

{

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

{

a++;

}

}

if (a == 0)

str++;

}

cout << "Kоличество строк, не содержащих ни одного нулевого элемента: " << str<<endl;//выводим количество строк

//инициализируем переменные, которые будут находить и считать наиболее часто встречающийся элемент матрицы

int maxCount = 0, maxItem = 0;

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

{

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

{

int count = 0;

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

{

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

{

if (A[i][j] == A[p][q])

{

count++;

}

}

}

if (count > maxCount)

{

maxCount = count;

maxItem = A[i][j];

}

}

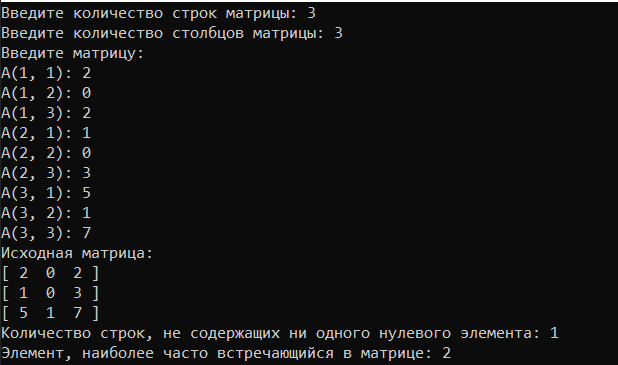
}

cout << "Элемент, наиболее часто встречающийся в матрице: " << maxItem << endl;//выводим элемент

delete[] A;//освобождение динамической памяти

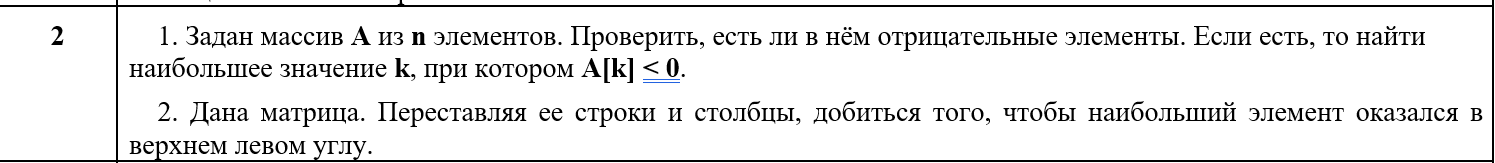
}

**Результат программы:**

****

**Дополнительные задания**

**Вариант 2**

****

**Задание 1**

**Код программы:**

#include <iostream>//подключение библиотеки ввода/вывода

using namespace std;//чтобы не писать каждый раз std

int main()//объявляем главную функцию

{

setlocale(LC\_ALL, "RU");//подключение кириллицы

//вводим размер массива

int n;

cout << "Введите размер массива: ";

cin >> n;

int\* A = new int[n];//создаем динамический массив

//вводим элементы массива

cout << "Введите элементы массива: ";

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

{

cin >> A[i];

}

//инициализируем переменные и флаги для поиска отрицательных элементов и их наибольшего индекса

bool otr = false;

int otr1 = -1;

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

{

if (A[i] < 0) {

otr = true;

otr1 = i;

}

}

if (otr1) {

cout << "В этом массиве есть отрицательные элементы" << endl;//выводим результат

cout << "Наибольший индекс: " << otr1 << endl;//выводим результат

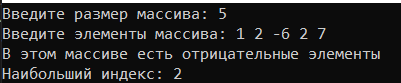
}

delete[] A;//очищаем массив из памяти

return 0;//завершение функции

}

**Результат программы:**

****

**Задание 2**

**Код программы:**

#include <iostream>//подключаем библиотеку ввода/вывода

using namespace std;//чтобы не писать каждый раз std

int main()//объявляем главную функцию

{

setlocale(LC\_ALL, "RU");//подключаем кириллицу

srand(time(NULL));//подключаем рандомайзер

//инициализируем переменные для размера массива, матрицы, максимальных значений и мест расположения в матрице

const int n = 3, m = 3;

int A[n][m];

int max1 = 0, row1 = 0, column1 = 0;

//используем цикл для генерирования элементов и поиска среди них 3 максимальных

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

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

A[i][j] = rand() % 100;

if (A[i][j] > max1) {

max1 = A[i][j]; row1 = i; column1 = j;

}

}

}

//перестановка элементов с помощью временной переменной

int temp = A[0][0];

A[0][0] = max1;

A[row1][column1] = temp;

temp = A[n / 2][m / 2];

//выводим новую матрицу

cout << "Получившаяся матрица:" << endl;

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

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

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

}

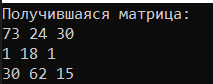
cout << endl;

}

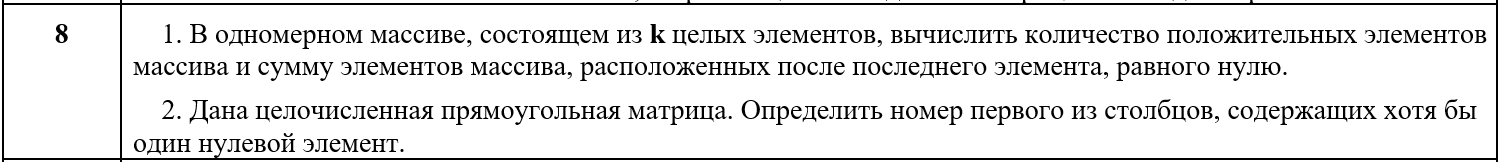
return 0;//завершаем программу

}

**Результат программы:**

****

**Вариант 8**

****

**Задание 1**

**Код программы:**

#include <iostream>//подключение библиотеки ввода/вывода

using namespace std;//чтобы не писать каждый раз std

int main()//объявляем главную функцию

{

setlocale(LC\_ALL, "RU");//подключаем кириллицу

//вводим длину массива

int k;

cout << "Введите длину массива: ";

cin >> k;

//создаем динамический массив

int\* A = new int[k];

//вводим элементы массива

cout << "Введите элементы массива: ";

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

{

cin >> A[i];

}

//инициализируем переменные для счетчика положительных элементов, поиска суммы элементов после 0, и переменную для поиска 0

int count = 0;

int lastzero = 0;

bool zero = false;

//с помощью цикла осуществляем поиск, сумму и подсчет

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

{

if (A[i] > 0) {

count++;

}

if (A[i] == 0) {

zero = true;

lastzero = 0;

}

if (zero) {

lastzero += A[i];

}

}

cout << "Количество положительных элементов: " << count<<endl;//выводим результат

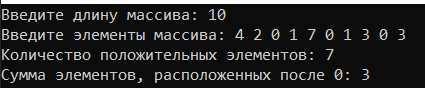
cout << "Сумма элементов, расположенных после 0: " << lastzero << endl;//выводим результат

delete[] A;//очищаем массив из памяти

return 0;//завершаем функцию

}

**Результат программы:**

****

**Примечание!!! Как Вы видите, программа посчитала сумму после последнего 0, а не после каждого, где берутся числа после каждого 0 и затем осуществляется сумма. Надеюсь, это не сильно повлияет на оценивание лабы😊**

**Задание 2**

**Код программы:**

#include <iostream>//подключение библиотеки ввода/вывода

using namespace std;//чтобы не писать каждый раз std

int main()//объявляем главную функцию

{

setlocale(LC\_ALL, "Russian");//подключаем кириллицу

//ининициализируем и вводим данные матрицы

int n, m;

cout << "Введите количество строк матрицы: ";

cin >> n;

cout << "Введите количество столбцов матрицы: ";

cin >> m;

int\*\* A = new int\* [n];//выделение динамической памяти

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

{

A[i] = new int[n];

}

cout << "Введите матрицу: " << endl;//вводим элементы матрицы

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

{

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

{

cout << "A(" << i + 1 << ", " << j + 1 << "): ";

cin >> A[i][j];

}

}

cout << "Исходная матрица: " << endl;//выводим исходную матрицу на экран

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

{

cout << "[";

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

{

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

}

cout << "]" << endl;

}

//инициализируем переменную, которая будет искать комер столбика, содержащего хотя бы 1 нулевой элемент

int stolb = 0;

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

{

int a = 0;

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

{

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

{

a++;

break;

}

}

if (a > 0)

{

stolb = i + 1;

break;

}

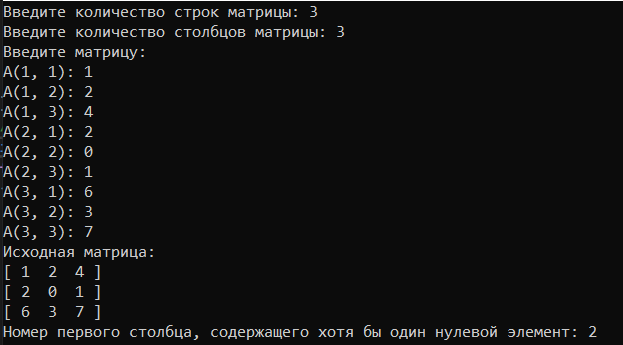
}

cout << "Номер первого столбца, содержащего хотя бы один нулевой элемент: " << stolb << endl;//вывод результата

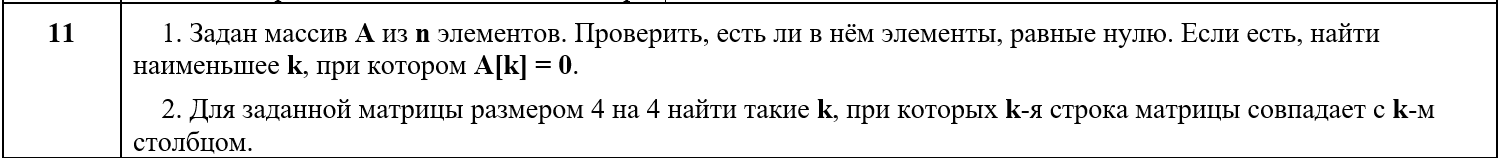
delete[] A;//освобождение динамической памяти

}

**Результат программы:**

****

**Вариант 11**

****

**Задание 1**

**Код программы:**

#include <iostream>//подключение библиотеки ввода/ввода

using namespace std;//чтобы не писать каждый раз std

int main()//объявляем главную функцию

{

setlocale(LC\_ALL, "RU");//подключение кириллицы

//вводим размер массива

int n;

cout << "Введите размер массива: ";

cin >> n;

int\* A = new int[n];//создаем динамического массива

cout << "Введите элементы массива: ";//вводим элементы массива

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

{

cin >> A[i];

}

//инициализация флагов и переменной для поиска элементов, равных 0 и наименьшего из индексов

bool zero = false;

int minIndex = n;

//с помощью цикла осуществляем поиск и сравнение

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

{

if (A[i] == 0 && i < minIndex) {

zero = true;

minIndex = i;

}

}

if (zero) {

cout << "В этом массиве есть элементы, равные нулю" << endl;//выводим результат

cout << "Минимальный индекс: " << minIndex << endl;//выводим результат

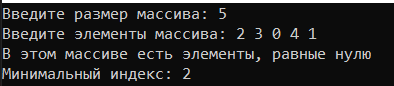
}

delete[] A;//очищаем массив из памяти

return 0;//завершение программы

}

**Результат программы:**

****

**Задание 2**

**Код программы:**

#include <iostream>//подключение библиотеки ввода/вывода

using namespace std;//чтобы не писать каждый раз std

int main()//объявляем главную функцию

{

setlocale(LC\_ALL, "RU");//подключение кириллицы

const int n = 4;//создание размеров матрицы

int matrix[n][n];//инициализация матрицы

//заполнение матрицы

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

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

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

cin >> matrix[i][j];

}

}

//с помощью цикла и флагов находим строки и столбцы с одинаковыми элементами

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

bool elements = true;

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

if (matrix[k][i] != matrix[i][k]) {

elements = false;

break;

}

}

if (elements) {

cout << "Строка " << k+1 << " совпадает со столбцом " << k+1 << endl;//вывод результата

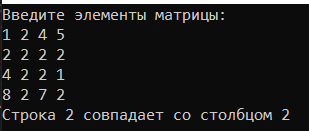
}

}

return 0;

}

**Результат программы:**

****