МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

#### Факультет экономики, менеджмента и информационных технологий

(факультет)

### Кафедра Систем управления и информационных технологий в строительств

# ОТЧЁТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

по дисциплине Объектно-ориентированное программирование

Тема «Работа с массивами»

Выполнил студент группы ИСТ-214 Д. В. Тюленев Подпись, дата Инициалы, фамилия

## Принял Е. Н. Королев Подпись, дата Инициалы, фамилия

Защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата

2022

Лабораторная работа № 2

«Работа с массивами»

Вариант №4

1. Цель работы: Научиться работать с массивами на языке С++
2. Задания на лабораторную работу:
   1. Дан одномерный массив, состоящий из N вещественных элементов.

* Заполнить массив случайными числами.
* Найти минимальный положительный элемент.
* Вычислить произведение не нулевых элементов массива.
* Вывести ненулевые элементы на экран в обратном порядке.
  1. Дан двумерный массив размером n m, заполненный случайными числами. Определить, есть ли в данном массиве столбец, в котором равное количество положительных и отрицательных элементов.
  2. Выполнить сортировку элементов одномерного массива используя метод «сортировки слиянием»

1. Ход выполнения:

Для начала необходимо объявить и реализовать функцию для генерации случайных чисел (далее в заданиях будет применяться) «lab2::random». Она не будет принимать параметров, но будет возвращать сгенерированное при помощи функции «rand» число вещественного типа.

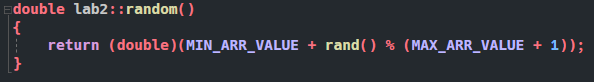


Рисунок 1 － Функция генерации числа в определённом интервале

Формула для определения границы генерации представляет собой следующие выражение: MIN + «функция генерации» % (MAX + 1) , где MIN － минимально возможное сгенерированное число (константное значение, задаётся при помощи директивы препроцессора －#define MIN\_ARR\_VALUE «число»), MAX - максимальное сгенерированное число (константное значение, задаётся при помощи директивы препроцессора －#define MAX\_ARR\_VALUE «число»).

* 1. Задание №1:

1. Для заполнения массива случайными числами определена функция «lab2::task1::generate\_arr», которая принимает в качестве параметров: «arr» －массив вещественного типа (указатель на первый элемент массива) и «n»－ размер массива целочисленного типа. Возвращает заполненный сгенерированными значениями массив. В теле функции открывается цикл для прохождения по всем элементам массива и установки сгенерированного значения путём вызова функции «lab2::random».

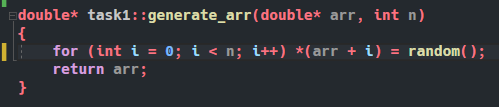


Рисунок 2 － Функция заполнения массива случайными числами

1. Для нахождения минимального положительного элемента определим функцию «lab2::task1::find\_min\_positive», которая принимает в качестве параметров: массив вещественного типа (указатель на первый элемент массива) и размер массива. Возвращает ссылку на найденный элемент массива.

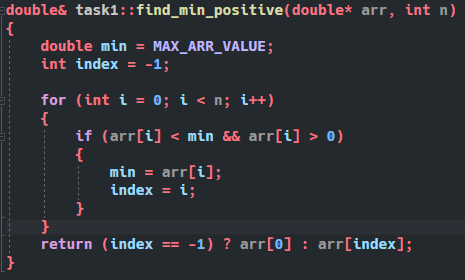


Рисунок 3 － Функция поиска минимального положительного числа

В начале инициализируется переменные «min» для хранения минимального найденного положительного числа, и «index» для хранения индекса найденного элемента. После открывается цикл для перебора значений массива. В теле цикла устанавливается условие: если элемент меньше минимального найденного и больше нуля, то обновляется значения переменных «min» и «index». В конце функции проверяется было ли найдено число путём сравнения текущего значения переменной «index» с начальным: если значение не изменилось － число не было найдено и возвращается ссылка на первый элемент массива, иначе － возвращается ссылка на найденный элемент.

1. Для вычисления произведения не нулевых элементов масса определена функция «lab2::task1::calculate», которая принимает в качестве параметров: массив вещественного типа (указатель на первый элемент массива) и размер массива. Возвращает рассчитанное произведение.

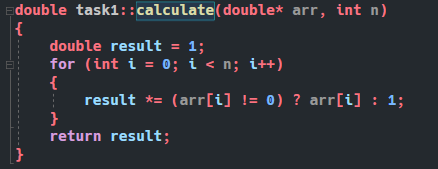


Рисунок 4 － Функция для вычисления произведения не нулевых элементов массива

Инициализируется переменная «result» отвечающая за хранение посчитанного произведения. Далее используется цикл для перебора элементов массива. В теле цикла через условный тернарный оператор проверяется является ли текущий взятый из массива элемент ненулевым, и далее, если утверждение верно, то значение элемента умножается на значение текущего вычисленного результата.

1. Для вывода ненулевых элементов в обратном порядке определена процедура «lab2::task1::print\_arr», которая принимает в качестве параметров: массив вещественного типа (указатель на первый элемент массива) и размер массива.

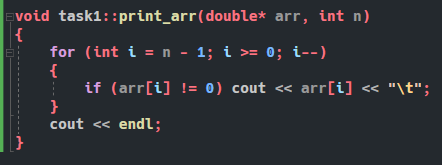


Рисунок 5 － Функция для вывода ненулевых элементов в обратном порядке

В теле процедуры открывается цикл for от «size - 1» до «0» с шагом «-1» для перебора элементов массива в обратном порядка. Внутри через условный оператор происходит проверка текущего выбранного элемента: если его значение отлично от нуля, происходит вывод значения в консоль.

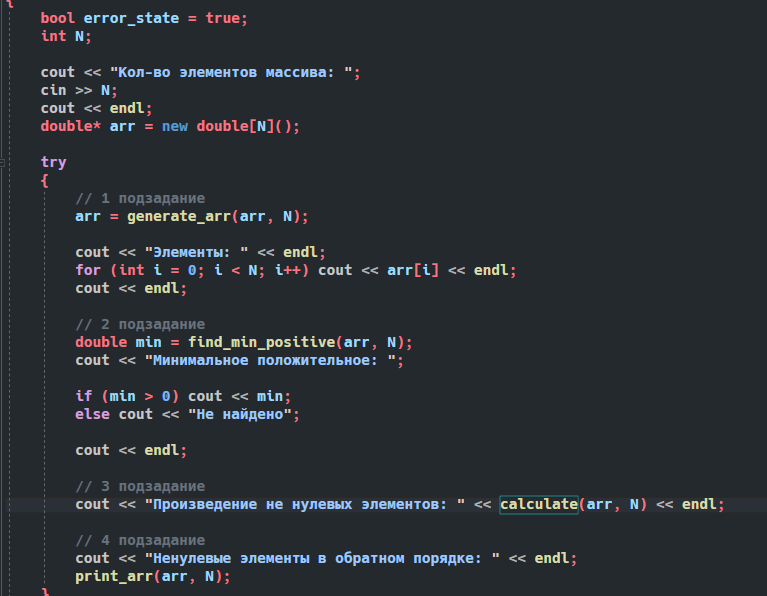


Рисунок 6 － Решение задания №1

В начале происходит объявление переменной «N» для хранения размера массива, далее происходит считывание значения из потока ввода. После с помощью оператора «new» инициализируется динамический массив.

Производится вызов функции «lab2::task1::generate\_arr» для генерации значений элементов, после чего она в теле цикла выводятся на экран.

Производится вызов функции «lab2::task1::find\_min\_positive» для поиска минимального положительного числа. Значение вызова записывается в переменную и проверяется: является ли оно положительным － значение выводится на экран, иначе － выводится специального сообщение о ошибке.

Производится вызов функции «lab2::task1::calculate» и вывод на экран возвращённое значение вызова.

Производится вызов функции «lab2::task1::print\_arr», которая выводит массив в обратном порядке, отбирая только ненулевые элементы.

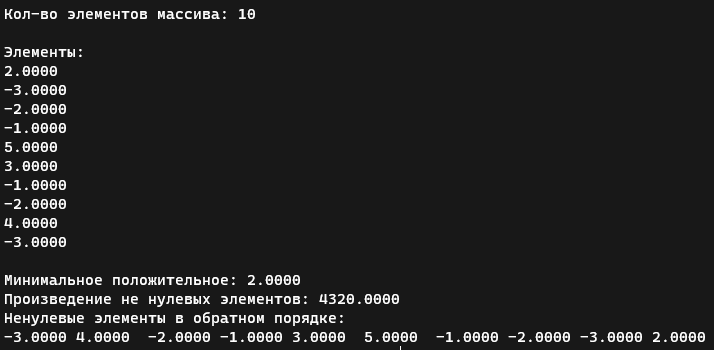


Рисунок 7 － Вывод задания №1

* 1. Задание №2:

Для генерации двумерного массива с заполнением случайными числами определяется функция «lab2::task2::generate\_matrix». Она принимает в качестве параметров размеры нового двумерного массива и возвращает созданную матрицу типа «task2::matrix\_t» (using matrix\_t = vector<vector<int>>). В теле функции объявлена переменная для хранения возвращаемого результата. После открывается внешний цикл для добавления строк матрицы через метод «push» объекта vector<vector>, далее открывается внутренний цикл, который наполняет строки значениями, рассчитанных при помощи функции «lab2::random».

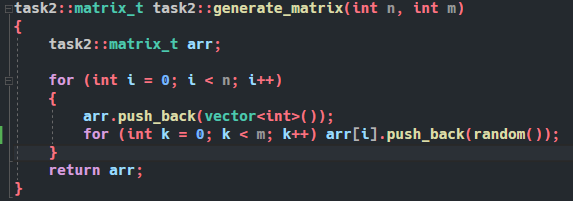


Рисунок 8 － Функция генерации двумерного массива

Для начала решения, необходимо произвести объявление переменных для хранения размеров массива, а после считать из консоли. Далее инициализируется динамический массив «counter» для хранения значений «смещения» для каждого столбца. «Смещение» позволяет определить сколько отрицательных и положительных чисел в столбце. Оно определяется следующим путём: если в строке значение для выбранного столбца отрицательное, то происходить уменьшение значения «смещения» для выбранного столбца на 1; иначе происходит увеличение на 1. Индексы «смещения» и столбцов совпадают. Если значение «смещения» равно нулю, то отрицательных числе и положительных равное количество.

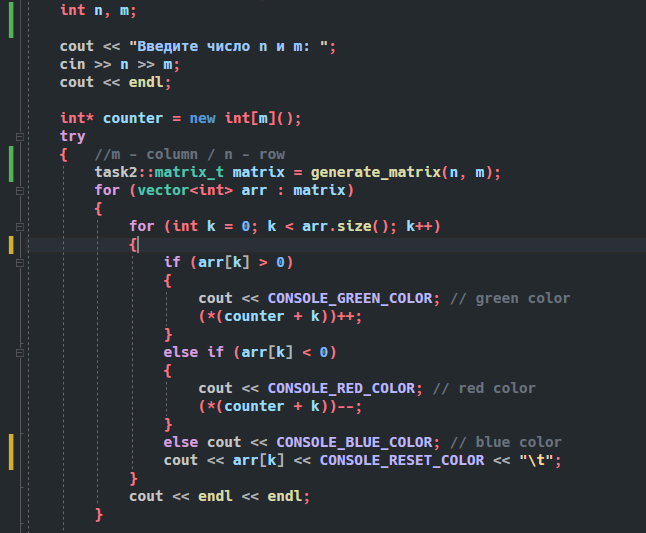


Рисунок 9 － Решение задания №2

После происходит вызов функции «lab2::task2::generate\_matrix» и возвращаемый результат записывается в переменную «matrix». Далее через вложенный цикл производится перебор строк и столбцов двумерного массива, для отображения на экране значений ячеек матрицы и вычисления «смещения» для каждого столбца.

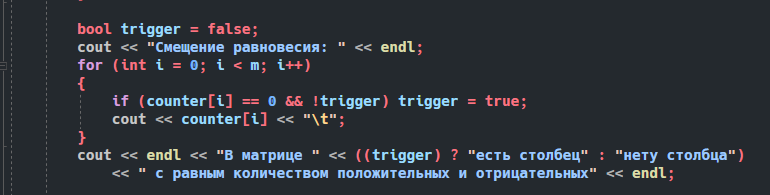


Рисунок 10 － Решение задания №2 (продолжение)

После инициализируется переменная «trigger» для хранения состояния утверждения «есть ли столбец с равным количеством...?». Далее происходит перевод значений массива «counter» в поиске элемента со значением «0». Через условный оператор каждый элемент проверяется и выводится на экран.

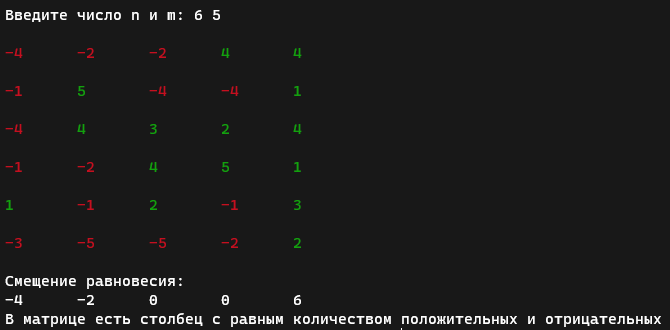


Рисунок 11 － Вывод задания №2

* 1. Задание №3:

Объявляется переменная «n» целочисленного типа для хранения размера массива и считывается значение в неё из потока. После создаётся динамический массив соответствующего размера. Открывается цикл для прохода по каждому элементу и установлению возращённого значения вызова функции «lab2::random» (генерации случайных чисел в диапазоне). После происходит вызов функции сортировки слиянием － «lab2::task3::sort\_merge». Возвращённый результат выводится на экран.

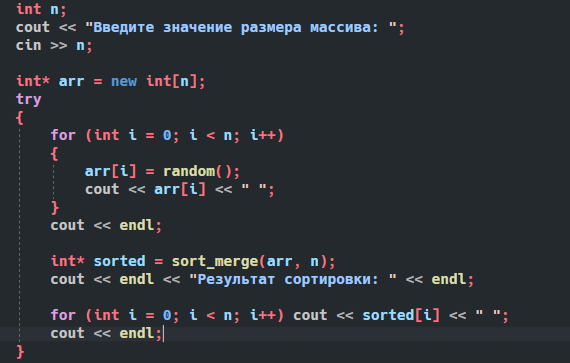


Рисунок 12－ Решение задания №3

Функция «lab2::task3::sort\_merge» принимает два параметра: «arr» －исходный массив (указатель на первый элемент) и «n» － размер массива. Возвращает отсортированный массив.

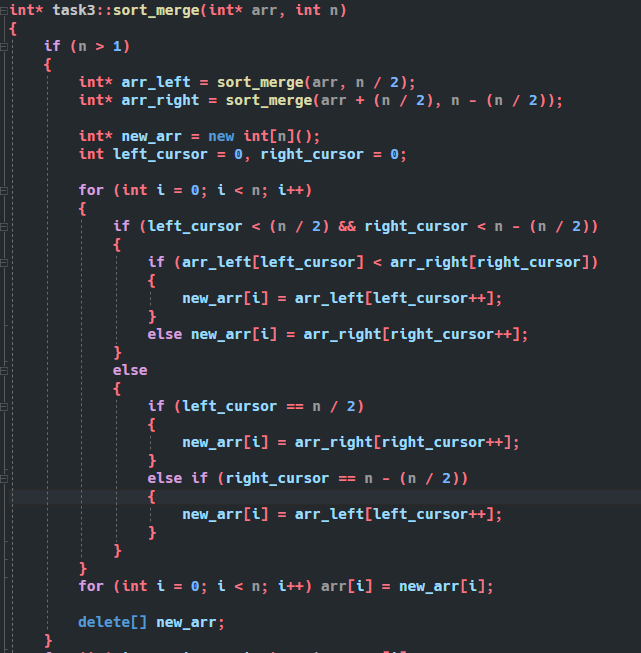


Рисунок 13 － Функция для сортировки слиянием.

В начале тела функции происходит проверка на размер массива: если массив меньше или равен«1», то значение массива при помощи оператора «return» возвращается; иначе － инициализируются переменные «arr\_left» (обработка левой части от текущего массива) и «arr\_right» (обработка правой части от текущего массива). Дважды совершается рекурсивный вызов: в первом в качестве параметра «arr» устанавливается указатель на начало текущего массива, и в качестве параметра размера устанавливается половина от текущего; во втором в качестве указателя на начало массива устанавливается адрес центрального элемента текущего массива, и в качестве размера массива － разность между размером текущего массива и массива отделённого для левой части.

После выделяется память на хранение динамического массива «new\_arr», инициализируются курсоры для левой и правой частей массива. Далее открывается цикл в котором проходим по всем элементам массива «new\_arr». В теле цикла совершается проверка на то, что достиг один из курсоров конца своей части при сортировке.

Если утверждение ложно, то сравниваются два элемента из разных частей, на которых установленны курсоры, и определяются какой будет установлен в качестве текущего элемента массива «new\_arr»; для части, из которой был взят элемент, будет двигаться курсор. Иначе, если утверждение ложно, то проверяется курсор какой части дошёл до конца, и в зависимости от этого элементы противоположной части будут до заполнять массив «new\_arr».

После этого значения массива «new\_arr» копируются в текущий массив, а память из под динамического высвобождается. После чего значение текущего возвращается из функции.

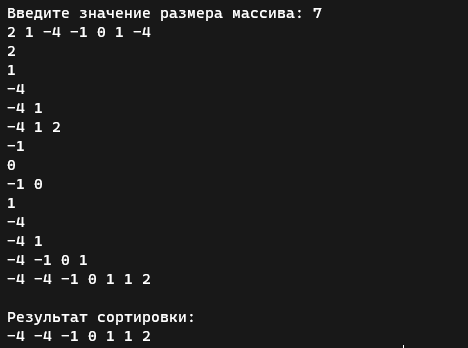


Рисунок 14 － Вывод задания №3

Вывод: научились работать с массивами на языке С++.

Приложение А

Листинг программного кода языка C++

Файл «lab2.h»

#pragma once

#include <iostream>

#include <locale>

#include <cmath>

#include <vector>

// для случайной генерации чисел в массив.

#define MAX\_ARR\_VALUE 10

#define MIN\_ARR\_VALUE -5

#define CONSOLE\_GREEN\_COLOR "\033[32m"

#define CONSOLE\_RED\_COLOR "\033[31m"

#define CONSOLE\_BLUE\_COLOR "\033[36m"

#define CONSOLE\_RESET\_COLOR "\033[0m"

namespace lab2

{

using namespace std;

double random();

namespace task1

{

double\* generate\_arr(double\* arr, int n);

double& find\_min\_positive(double\* arr, int n);

double calculate(double\* arr, int n);

void print\_arr(double\* arr, int n);

bool run(void);

}

namespace task2

{

using matrix\_t = vector<vector<int>>;

matrix\_t generate\_matrix(int n, int m);

bool run(void);

}

namespace task3

{

int\* sort\_merge(int\* arr, int n);

bool run(void);

}

void lab(void);

}

Файл «lab2.cpp»

#include "lab2.h"

using namespace lab2;

double\* task1::generate\_arr(double\* arr, int n)

{

for (int i = 0; i < n; i++) \*(arr + i) = random();

return arr;

}

double& task1::find\_min\_positive(double\* arr, int n)

{

double min = MAX\_ARR\_VALUE;

int index = -1;

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

{

if (arr[i] < min && arr[i] > 0)

{

min = arr[i];

index = i;

}

}

return (index == -1) ? arr[0] : arr[index];

}

double task1::calculate(double\* arr, int n)

{

double result = 1;

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

{

result \*= (arr[i] != 0) ? arr[i] : 1;

}

return result;

}

void task1::print\_arr(double\* arr, int n)

{

for (int i = n - 1; i >= 0; i--)

{

if (arr[i] != 0) cout << arr[i] << "\t";

}

cout << endl;

}

bool task1::run(void)

{

bool error\_state = true;

int N;

cout << "Кол-во элементов массива: ";

cin >> N;

cout << endl;

double\* arr = new double[N]();

try

{

// 1 подзадание

arr = generate\_arr(arr, N);

cout << "Элементы: " << endl;

for (int i = 0; i < N; i++) cout << arr[i] << endl;

cout << endl;

// 2 подзадание

double min = find\_min\_positive(arr, N);

cout << "Минимальное положительное: ";

if (min > 0) cout << min;

else cout << "Не найдено";

cout << endl;

// 3 подзадание

cout << "Произведение не нулевых элементов: " << calculate(arr, N) << endl;

// 4 подзадание

cout << "Ненулевые элементы в обратном порядке: " << endl;

print\_arr(arr, N);

}

catch (...)

{

error\_state = false;

}

delete[] arr;

return error\_state;

}

task2::matrix\_t task2::generate\_matrix(int n, int m)

{

task2::matrix\_t arr;

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

{

arr.push\_back(vector<int>());

for (int k = 0; k < m; k++) arr[i].push\_back(random());

}

return arr;

}

bool task2::run(void)

{

bool error\_state = true;

int n, m;

cout << "Введите число n и m: ";

cin >> n >> m;

cout << endl;

int\* counter = new int[m]();

try

{ //m - column / n - row

task2::matrix\_t matrix = generate\_matrix(n, m);

for (vector<int> arr : matrix)

{

for (int k = 0; k < arr.size(); k++)

{

if (arr[k] > 0)

{

cout << CONSOLE\_GREEN\_COLOR; // green color

(\*(counter + k))++;

}

else if (arr[k] < 0)

{

cout << CONSOLE\_RED\_COLOR; // red color

(\*(counter + k))--;

}

else cout << CONSOLE\_BLUE\_COLOR; // blue color

cout << arr[k] << CONSOLE\_RESET\_COLOR << "\t";

}

cout << endl << endl;

}

bool trigger = false;

cout << "Смещение равновесия: " << endl;

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

{

if (counter[i] == 0 && !trigger) trigger = true;

cout << counter[i] << "\t";

}

cout << endl << "В матрице " << ((trigger) ? "есть столбец" : "нету столбца")

<< " с равным количеством положительных и отрицательных" << endl;

}

catch (...)

{

error\_state = false;

}

delete[] counter;

return error\_state;

}

int\* task3::sort\_merge(int\* arr, int n)

{

if (n > 1)

{

int\* arr\_left = sort\_merge(arr, n / 2);

int\* arr\_right = sort\_merge(arr + (n / 2), n - (n / 2));

int\* new\_arr = new int[n]();

int left\_cursor = 0, right\_cursor = 0;

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

{

if (left\_cursor < (n / 2) && right\_cursor < n - (n / 2))

{

if (arr\_left[left\_cursor] < arr\_right[right\_cursor])

{

new\_arr[i] = arr\_left[left\_cursor++];

}

else new\_arr[i] = arr\_right[right\_cursor++];

}

else

{

if (left\_cursor == n / 2)

{

new\_arr[i] = arr\_right[right\_cursor++];

}

else if (right\_cursor == n - (n / 2))

{

new\_arr[i] = arr\_left[left\_cursor++];

}

}

}

for (int i = 0; i < n; i++) arr[i] = new\_arr[i];

delete[] new\_arr;

}

for (int i = 0; i < n; i++) cout << arr[i] << " ";

cout << endl;

return arr;

}

bool task3::run(void)

{

bool error\_state = true;

int n;

cout << "Введите значение размера массива: ";

cin >> n;

int\* arr = new int[n];

try

{

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

{

arr[i] = random();

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

}

cout << endl;

int\* sorted = sort\_merge(arr, n);

cout << endl << "Результат сортировки: " << endl;

for (int i = 0; i < n; i++) cout << sorted[i] << " ";

cout << endl;

}

catch (...)

{

error\_state = false;

}

delete[] arr;

return error\_state;

}

double lab2::random()

{

return (double)(MIN\_ARR\_VALUE + rand() % (MAX\_ARR\_VALUE + 1));

}

void lab2::lab(void)

{

cout << fixed;

cout.precision(4);

bool exit\_trigger = false, error\_handler = true;

srand(time(0));

while (1)

{

system("cls");

int task\_number = -1;

cout << endl

<< "1 - Проверка Задачи 1\n"

<< "2 - Проверка Задачи 2\n"

<< "3 - Проверка Задачи 3\n"

<< "4 - Выход из программы" << endl << endl;

cout << "Введите номер задания: " << endl;

cin >> task\_number;

system("cls");

switch (task\_number)

{

case 1:

error\_handler = task1::run();

break;

case 2:

error\_handler = task2::run();

break;

case 3:

error\_handler = task3::run();

break;

case 4:

exit\_trigger = true;

break;

default:cout << "Неверный ввод" << endl;

}

if (exit\_trigger) break;

else

{

if (!error\_handler) cout << "При выполнении задания возникла ошибка\n" << endl;

}

system("pause");

}

}

Файл «main.cpp»

#include <iostream>

#include <locale>

#include "labs/lab2/lab2.h"

int main(void)

{

setlocale(LC\_ALL, "rus");

lab2::lab();

return 0;

}