Федеральное государственное бюджетное образовательное учреждение высшего образования "Национальный исследовательский университет "МЭИ"

**ОТЧЕТ**

**По лабораторной работе № 3**

**«Функции и структуры»**

ФИО студента Аносова Ольга

Номер группы А-01-18

Имя преподавателя Мохов Андрей Сергеевич

Василькова Полина Денисовна

**Цель работы**

1. Уметь структурировать программу при помощи функций и структур.
2. Уметь писать модульные тесты.

# Часть 1. Декомпозиция программы функциями

## Задание

Берем исходный код программы для ЛР 1 и разделяем его на части в функции:

* Ввод чисел:
  + принимает количество чисел, которое необходимо ввести;
  + возвращает вектор чисел.
* Поиск наибольшего и наименьшего значения:
  + принимает вектор чисел;
  + возвращает два результата — min и max.
* Расчет гистограммы:
  + принимает вектор чисел и количество корзин;
  + возвращает вектор количеств чисел в каждой корзине;
  + *вызывает* в процессе работы функцию поиска min и max.

# Часть 2. Вывод гистограммы как изображения в формате SVG

Изображения в формате SVG представляют собой текстовые файлы, в которых перечислены геометрические фигуры (линии, прямоугольники и прочие) и их свойства (положение, размер, цвет и другие).

Задание

Переделать вывод гистограммы в формат SVG.

Весь вывод гистограммы в SVG реализован в функции show\_histogram\_svg(). Сначала выводится пустое изображение фиксированного размера. Затем записываются функции вывода элементов SVG. Цвета столбцов гистограммы могут быть изменены.

# Часть 3. Модульное тестирование

Пишем модульный тест для функции поиска минимума и максимума, то есть Функцию нахождения минимума и максимума выделяем в отдельный файл histogram.cpp, который в свою очередь добавляем в файл histogram.h, который инициализирован в основной программе main.cpp командой #include “histogram.h”.

**Создание проекта для модульных тестов**

Создаем новый проект lab03-test, добавляем к нему файлы histogram.h и histogram.cpp. В функции main проекта пишем тесты для функции нахождения минимума и максимума, такие как случай положительных чисел, отрицательных чисел и так далее. Для этого используется библиотека <casset> и функция assert(). Если скомпилировать проект lab03-test, то программа прогонит все тесты и выдаст положительный результат, который будет означать успешное прохождение тестов, но не обязательно отсутствие ошибок в исходном коде, либо неуспешное завершение тестов, что будет означать, что есть ошибки в исходном коде, и что какой то тест не был пройден, так же будет указано какой именно.

**КОД main.cpp**

#include "svg.h"

#include "hist.h"

#include <iostream>

#include <vector>

#include <math.h>

using namespace std;

vector<double>

input\_numbers(size\_t count)

{

vector<double> result(count);

for (size\_t i = 0; i < count; i++)

{

cin >> result[i];

}

return result;

}

vector <size\_t> make\_histogram(vector<double> numbers, size\_t bin\_count)

{

double min, max;

find\_minmax(numbers, min, max);

vector<size\_t> bins(bin\_count);

for (double number : numbers)

{

size\_t bin = (size\_t)((number - min) / (max - min) \* bin\_count);

if (bin == bin\_count)

{

bin--;

}

bins[bin]++;

}

return bins;

}

int show\_histogram\_text(vector<size\_t> bins)

{

const size\_t SCREEN\_WIDTH = 80;

const size\_t MAX\_ASTERISK = SCREEN\_WIDTH - 4 - 1;

size\_t max\_count = 0;

for (size\_t count : bins)

{

if (count > max\_count)

{

max\_count = count;

}

}

const bool scaling\_needed = max\_count > MAX\_ASTERISK;

for (size\_t bin : bins)

{

if (bin < 100)

{

cout << ' ';

}

if (bin < 10)

{

cout << ' ';

}

cout << bin << "|";

size\_t height = bin;

if (scaling\_needed)

{

const double scaling\_factor = (double)MAX\_ASTERISK / max\_count;

height = (size\_t)(bin \* scaling\_factor);

}

for (size\_t i = 0; i < height; i++)

{

cout << '\*';

}

cout << '\n';

}

return 0;

}

int main()

{

// ¬вод данных

size\_t number\_count;

cerr << "Enter number count: ";

cin >> number\_count;

cerr << "Enter numbers: ";

const auto numbers = input\_numbers(number\_count);

size\_t bin\_count;

cerr << "Enter column count: ";

cin >> bin\_count;

if (bin\_count == 0)

bin\_count = sqrt(number\_count);

if (bin\_count > 25)

bin\_count = 1 + log2(number\_count);

// ќбработка данных

const vector<size\_t> bins = make\_histogram(numbers, bin\_count);

// ¬ывод данных

show\_histogram\_svg(bins);

//show\_histogram\_svg(bins);

}

else for (size\_t j = 0; j < bins[i]; j++)

{

cout << "\*";

}

cout << "\n";

}

system("pause");

}

**kod\_svg\_h.txt**

#ifndef SVG\_H\_INCLUDED

#define SVG\_H\_INCLUDED

#include <vector>

#include <iostream>

using namespace std;

void

svg\_begin(double width, double height)

{

cout << "<?xml version='1.0' encoding='UTF-8'?>\n";

cout << "<svg width='" << width << "' height='" << height<< "' "

<< "viewBox='0 0 " << width << " " << height << "' "

<< "xmlns='http://www.w3.org/2000/svg'>\n";

}

void

svg\_end()

{

cout << "</svg>\n";

}

void svg\_text(double left, double baseline, string text){

cout <<"<text x = '"<<left<<"' y = '"<<baseline<<"' >"<< text <<"</text>";

}

void svg\_rect(double x, double y, double width, double height,

string stroke = "black", string fill = "black"){

cout << "<rect x= '"<<x<<"' y= '"<<y<<"' width= '"<<width<<"' height= '"<<height<<"' stroke= '"<<stroke<<"' fill= '"<<fill<<"' />";

}

void

show\_histogram\_svg(const vector<size\_t>& bins) {

const auto IMAGE\_WIDTH = 400;

const auto IMAGE\_HEIGHT = 300;

const auto TEXT\_LEFT = 20;

const auto TEXT\_BASELINE = 20;

const auto TEXT\_WIDTH = 50;

const auto BIN\_HEIGHT = 30;

double top = 0;

svg\_begin(400, 300);

for (size\_t bin : bins)

{

const double bin\_width = 10 \* bin;

svg\_text(TEXT\_LEFT, top + TEXT\_BASELINE, to\_string(bin));

svg\_rect(TEXT\_WIDTH, top, bin\_width, BIN\_HEIGHT, "blue", "yellow");

top += BIN\_HEIGHT;

}

svg\_end();

}

#endif // SVG\_H\_INCLUDED

int main()

{

// ¬вод данных

size\_t number\_count;

cerr << "Enter number count: ";

cin >> number\_count;

cerr << "Enter numbers: ";

const auto numbers = input\_numbers(number\_count);

size\_t bin\_count;

cerr << "Enter column count: ";

cin >> bin\_count;

// ќбработка данных

const vector<size\_t> bins = make\_histogram(numbers, bin\_count);

// ¬ывод данных

show\_histogram\_svg(bins);

//show\_histogram\_svg(bins);

}

**histogram\_cpp.txt**

#include "hist.h"

void

find\_minmax(vector<double> numbers, double& min, double& max)

{

if (numbers.size() != 0)

{

min = numbers[0];

max = numbers[0];

for (double number : numbers)

{

if (max < number)

max = number;

if (min > number)

min = number;

}

}

}

#include "hist.h"

#include <cassert>

void

test\_positive()

{

double min = 0;

double max = 0;

find\_minmax({1, 2, 3}, min, max);

assert(min == 1);

assert(max == 3);

}void test\_negative()

{

double min = 0;

double max = 0;

find\_minmax({-1, -2, -3}, min, max);

assert (min == -3);

assert (max == -1);

}

void test\_odin()

{

double min = 0;

double max = 0;

find\_minmax({5}, min, max);

assert (min == 5);

assert (max == 5);

}

void test\_pustoy()

{

double min = 0;

double max = 0;

}

int

main()

{

test\_positive();

test\_negative();

test\_odin();

test\_pustoy();

}#ifndef HIST\_H\_INCLUDED

#define HIST\_H\_INCLUDED

#include <vector>

using namespace std;

void

find\_minmax(vector<double> numbers, double& min, double& max);

#endif // HIST\_H\_INCLUDED

**test\_cpp.txt**

#include "hist.h"

#include <cassert>

void

test\_positive()

{

double min = 0;

double max = 0;

find\_minmax({1, 2, 3}, min, max);

assert(min == 1);

assert(max == 3);

}

void test\_negative()

{

double min = 0;

double max = 0;

find\_minmax({-1, -2, -3}, min, max);

assert (min == -3);

assert (max == -1);

}

void test\_odin()

{

double min = 0;

double max = 0;

find\_minmax({5}, min, max);

assert (min == 5);

assert (max == 5);

}

void test\_pustoy()

{

double min = 0;

double max = 0;

}

int

main()

{

test\_positive();

test\_negative();

test\_odin();

test\_pustoy();

}