Федеральное государственное бюджетное образовательное учреждение высшего образования «Национальный исследовательский университет «МЭИ»

Институт информационных и вычислительных технологий

Кафедра Управления и интеллектуальных технологий

**Отчёт по лабораторной работе № 3**

**По курсу «Разработка программного обеспечения систем управления»**

# «Декомпозиция программы»

Выполнил студент группы А-03-20

Коротков Егор Дмитриевич

Проверил

Мохов А.С.

Козлюк Д.А.

Василькова П.Д.

Москва 2021

Цель работы:

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

Код файла main.cpp:

#include <iostream>

#include <vector>

#include <iomanip>

#include "histogram.h"

using namespace std;

void show\_histogram\_text(const 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';

}

}

vector<double> input\_numbers(size\_t count)

{

vector<double> result(count);

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

{

cin >> result[i];

}

return result;

}

int make\_histogram(const vector<double>& numbers, size\_t bin\_count, vector<size\_t>& bins)

{

double min, max;

find\_minmax(numbers, min, max);

for (double number : numbers)

{

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

if (bin == bin\_count)

{

bin--;

}

bins[bin]++;

}

}

int main()

{

// Ввод данных

size\_t number\_count;

cerr << "Enter number count: ";

cin >> number\_count;

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

size\_t bin\_count;

cerr << "Enter column count: ";

cin >> bin\_count;

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

make\_histogram(numbers, bin\_count, bins);

show\_histogram\_svg(bins);

return 0;

}

Код файла histogram.cpp:

#include "histogram.h"

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

min = numbers[0];

max = numbers[0];

for (double number : numbers) {

if (number < min) {

min = number;

}

if (number > max) {

max = number;

}

}

}

void svg\_rect(double x, double y, double width, double height, string stroke, string fill)

{

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;

const auto BLOCK\_WIDTH = 10;

double top = 0;

const size\_t MAX\_ASTERISK = IMAGE\_HEIGHT - TEXT\_BASELINE \* 2;

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;

svg\_begin( 300, 400);

for (size\_t bin : bins)

{

size\_t height = bin;

if (scaling\_needed)

{

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

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

}

const double bin\_width = BIN\_HEIGHT \* height;

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

svg\_rect(top + TEXT\_LEFT, TEXT\_BASELINE \* 2, BIN\_HEIGHT, bin\_width, "black", "blue");

top += BIN\_HEIGHT ;

}

svg\_end();

}

void svg\_end()

{

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

}

void svg\_begin(double width, double height)

{

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

cout << "<svg ";

cout << "width='" << width << "' ";

cout << "height='" << height << "' ";

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

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

}

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

{

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

}

Код файла histogram.h:

#ifndef HISTOGRAM\_H\_INCLUDED

#include <vector>

#include <iomanip>

#include <iostream>

#define HISTOGRAM\_H\_INCLUDED

using namespace std;

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

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

void svg\_end();

void svg\_begin(double width, double height);

void svg\_rect(double x, double y, double width, double height,string stroke, string fill);

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

#endif // HISTOGRAM\_H\_INCLUDED

Код файла test.cpp:

#include "histogram.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\_equally()

{

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 1);

}

void test\_alone()

{

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 1);

}

void test\_none()

{

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 2);

}

int main()

{

test\_positive();

test\_negative();

test\_equally();

test\_alone();

test\_none();

}

Вариант 10.

Задание варианта: Отображайте гистограмму вертикально, с подписями сверху, по аналогии с заданием этого варианта в лабораторной работе 1. Предусмотреть расчет IMAGE\_HEIGHT таким образом, чтобы вся гистограмма вмещалась в область рисунка.

Разработка.

Чтобы перевернуть гистограмму, используем функцию svg\_rect(top + TEXT\_LEFT, TEXT\_BASELINE \* 2, BIN\_HEIGHT, bin\_width, "black", "blue"), где top + TEXT\_LEFT - отступ от края по x, top в свою очередь переменная, которая с каждым циклом увеличивается на ширину одной “колонки” (BIN\_HEIGHT) гистограммы, константа TEXT\_BASELINE \* 2 - значение координаты нижнего края по вертикали (по y). За высоту одной “колонки” гистограммы принимаем значение bin\_width = BIN\_HEIGHT \* bin, где bin переменная по bins, а за ширину “колонки” берем константу BIN\_HEIGHT. Что касается теста, то текст выводим в строчку при помощи функции svg\_text(top + BIN\_HEIGHT, TEXT\_BASELINE , to\_string(bin)), где top + BIN\_HEIGHT - значение по x, а значение по y - константа TEXT\_BASELINE.

В случае, если максимальное значение bin будет больше IMAGE\_HEIGHT, то используем масштабирование.

if (scaling\_needed)

{

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

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

}

Где scaling\_needed (const bool scaling\_needed = max\_count > MAX\_ASTERISK) , а max\_count - максимальное значение bin, а height - высота столбца при масштабировании.