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»**

# «Основы языка С++»

Выполнила студентка группы А-01-20

Яблокова Валентина Андреевна

Проверил

Мохов А. С

Козлюк Д.А.

Москва 2021

Ссылка на Github: https://github.com/kisa29v/lab03

Код Лабораторной работы №3:

main.cpp

#include <iostream>

#include <vector>

#include "histogram.h"

using namespace std;

const size\_t SCREEN\_WIDTH = 80;

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

vector<double>

input\_numbers(size\_t count) //определение функции для ввода массива чисел

{

vector<double> result(count);

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

{

cin >> result[i];

}

return result;

}

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\_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;

const auto BLOCK\_WIDTH = 10;

svg\_begin(IMAGE\_WIDTH, IMAGE\_HEIGHT);

svg\_text(TEXT\_LEFT, TEXT\_BASELINE, to\_string(bins[0]));

svg\_rect(TEXT\_WIDTH, 0, bins[0] \* BLOCK\_WIDTH, BIN\_HEIGHT, "red", "#ffeeee");

double top = 0;

for (size\_t bin : bins)

{

const double bin\_width = BLOCK\_WIDTH \* bin;

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

svg\_rect(TEXT\_WIDTH, top, bin\_width, BIN\_HEIGHT, "red", "#ffeeee");

top += BIN\_HEIGHT;

}

svg\_end();

}

int main()

{

size\_t number\_count;

cerr << "Enter number count: ";

cin >> number\_count;

// Ввод чисел заменен вызовом функции:

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

size\_t bin\_count;

cerr << "Enter column count: ";

cin >> bin\_count;

const auto bins = make\_histogram(numbers, bin\_count);

show\_histogram\_svg(bins);

return 0;

}

histogram.h:

#ifndef HISTOGRAM\_H\_INCLUDED

#define HISTOGRAM\_H\_INCLUDED

#include <vector>

using namespace std;

void

find\_minmax(const vector<double>& numbers, double& min, double& max); //функция поиска максимума и минимума

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

#endif

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;

}

}

}

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

{

double min = numbers[0];

double max = numbers[0];

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;

}

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\_odinakov() {

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 1);

}

void

test\_odno() {

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 1);

}

void

test\_pysto() {

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 1);

}

int

main() {

test\_positive();

test\_negative();

}

Вариант 6

#include <iostream>

#include <vector>

#include "histogram.h"

using namespace std;

const size\_t SCREEN\_WIDTH = 80;

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

vector<double>

input\_numbers(size\_t count)

{

vector<double> result(count);

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

{

cin >> result[i];

}

return result;

}

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\_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, string z, 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;

const auto BLOCK\_WIDTH = 10;

svg\_begin(IMAGE\_WIDTH, IMAGE\_HEIGHT);

svg\_text(TEXT\_LEFT, TEXT\_BASELINE, to\_string(bins[0]));

string color\_colon;

svg\_rect(TEXT\_WIDTH, "pink", 0, bins[0] \* BLOCK\_WIDTH, BIN\_HEIGHT, "red", "#ffeeee");

double top = 0;

for (size\_t bin : bins)

{

const double bin\_width = BLOCK\_WIDTH \* bin;

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

svg\_rect(TEXT\_WIDTH, color\_colon, top, bin\_width, BIN\_HEIGHT, "red", "#ffeeee");

top += BIN\_HEIGHT;

}

svg\_end();

}

int main()

{

size\_t number\_count;

cerr << "Enter number count: ";

cin >> number\_count;

// Ввод чисел заменен вызовом функции:

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

size\_t bin\_count;

cerr << "Enter column count: ";

cin >> bin\_count;

const auto bins = make\_histogram(numbers, bin\_count);

show\_histogram\_svg(bins);

return 0;

}