**Отчет по Лабораторной работе №3**

*Студент* : Красночуб Глеб

*Группа* : А-01-18

*Преподаватели* : Мохов Андрей Сергеевич

Василькова Полина Денисовна

**Цель работы**

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

В ходе данной работы нам необходимо перенести используемые функции в отдельный файл. Для этого мы создаем заголовочный файл histogram.h, в котором находятся прототипы функций, и файл реализации histogram.cpp, в котором находятся определения функций. Файл histogram.h мы подключаем в основную программу и в файл histogram.cpp с помощью команды #include “histogram.h”.

Также в ходе лабораторной работы мы создали отдельный проект lab03-test, добавили к нему пустой файл test.cpp, а также файлы histogram.h и histogram.cpp. В файле test.cpp мы расписываем модульные тесты, которые нужны для выявления возможных ошибок.

**Код main.cpp**

#include <iostream>

#include <vector>

#include "histogram.h"

using namespace std;

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

{

cerr << "Enter numbers:\n";

vector<double> result(count);

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

{

cin >> result[i];

}

return result;

}

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;

// Обработка данных

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

// Вывод данных

//show\_histogram\_text(bins);

show\_histogram\_svg(bins);

return 0;

}

**Код histogram.cpp**

#ifndef HISTOGRAM\_H\_INCLUDED

#define HISTOGRAM\_H\_INCLUDED

#include<iostream>

#include <vector>

using namespace std;

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

{

if(numbers.size()==0) return;

min = numbers[0];

max = numbers[0];

for (double number : numbers)

{

if (number < min)

{

min = number;

}

if (number > max)

{

max = number;

}

}

}

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>";

//<text x="20" y="35">anything you want</text>

}

void svg\_rect(double x=0, double y=0, double width=100, double height=200,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);

string stroke;

string fill;

for (size\_t bin : bins)

{

const double bin\_width = 10 \* bin;

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

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

top += BIN\_HEIGHT;

}

svg\_end();

}

vector <size\_t> make\_histogram(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 = (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';

}

}

#endif // HISTOGRAM\_H\_INCLUDED

**Код histogram.h**

#ifndef HISTOGRAM\_H\_INCLUDED

#define HISTOGRAM\_H\_INCLUDED

#include <iostream>

#include <vector>

using namespace std;

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

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

void svg\_end();

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

void svg\_rect(double x=0, double y=0, double width=100, double height=200,string stroke="black",string fill="black");

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

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

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

#endif // HISTOGRAM\_H\_INCLUDED

**Код 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\_const() {

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 1);

}

void test\_one() {

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 1);

}

void test\_empty() {

double min = 0;

double max = 0;

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

assert(min == 0);

assert(max == 0);

}

int main() {

test\_positive();

test\_negative();

test\_const();

test\_one();

test\_empty();

}