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-02-20

Жильцов Александр Иванович

Проверил

Мохов А.С.

Козлюк Д.А.

Василькова П.Д.

Москва 2021

Ссылка на репозитарий: https://github.com/Q1ex/cs-lab03

**Цель работы:**

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

# Задание:

# Часть 1. Декомпозиция программы функциями

Программа для построения гистограммы из ЛР № 1 состоит из одной функции main() на более чем 100 строк, из-за чего в ней неудобно ориентироваться. Необходимо выделить части программы 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

Требуется вместо текстовой гистограммы рисовать картинку, например:

Изображение выглядит как стрела

Автоматически созданное описание

# Часть 3. Модульное тестирование

Написать модульный тест для функции поиска минимума и максимума.

**Индивидуальное задание. Вариант 10**

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.

**Логика решения варианта:**

Создаём функцию, которая будет рассчитывать размер IMAGE\_HEIGHT. И используя полученное значение строить гистограмму вертикально.

**Тесты для индивидуального задания:**

Тесты написаны для функции, которая вычисляет IMAHE\_HEIGHT . 1) тест, когда функции подаётся пустой массив, функции должна вывести 0 2) тест, когда функции подаётся массив из 1 элемента, ожидается, что полученное значение равно 80.

**Исходный код всех модулей:**

**main.cpp**

#include <iostream>

#include <iostream>

#include <vector>

#include "histogram.h"

#include "svg.h"

using namespace std;

const size\_t SCREEN\_WIDTH = 80;

const size\_t MAX\_ASTERISK = SCREEN\_WIDTH - 3 - 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;

}

vector<size\_t>

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

double min, max;

find\_minmax(numbers,min,max);

double bin\_size = (max-min)/bin\_count;

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

size\_t number\_count = numbers.size();

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

{

bool found = false;

for (size\_t j = 0; (j < bin\_count - 1) && !found; j++)

{

auto lo = min + j \* bin\_size;

auto hi = min + (j + 1) \* bin\_size;

if ((lo <= numbers[i]) && (numbers[i] < hi))

{

bins[j]++;

found = true;

}

}

if (!found)

{

bins[bin\_count - 1]++;

}

}

return bins;

}

void

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

size\_t max\_count = bins[0];

for (size\_t x : bins)

if (x > max\_count)

max\_count = x;

size\_t bin\_count = bins.size();

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

{

for (size\_t j = 0; j < bin\_count; j++)

{

if (bins[j] > i)

cout << "\*";

else cout << " ";

}

cout << endl;

}

}

int main()

{

size\_t number\_count;

cerr << "Enter number count: ";

cin >> number\_count;

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

size\_t bin\_count;

cerr << "bin\_count: ";

cin >> bin\_count;

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

//show\_histogram\_text(bins);

show\_histogram\_svg(bins);

}

#include "histogram.h"

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

{

if (numbers.size() == 0)

return;

else

{

min = numbers[0];

max = numbers[0];

for (double number : numbers)

{

if (number < min)

{

min = number;

}

if (number > max)

{

max = number;

}

}

}

}

double mean(const vector<size\_t>& bins) {

if (bins.size()==0)

return 0;

double result;

double sum=0;

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

{

sum = sum + (double)(bins[i]);

}

result=sum/((double)(bins.size()));

return result;

}

**histogram.cpp**

#include "histogram.h"

void

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

if(!numbers.size()==0){

min = numbers[0];

max= numbers[0];

for (double x : numbers)

{

if (x < min) min = x;

else if (x > max) max = x;

}

}

}

**histogram.h**

#ifndef HISTOGRAM\_H\_INCLUDED

#define HISTOGRAM\_H\_INCLUDED

#include <vector>

using namespace std;

void

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

#endif // HISTOGRAM\_H\_INCLUDED

**svg.cpp**

#include "svg.h"

#include "histogram.h"

#include <iostream>

using namespace std;

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, string fill){

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

}

size\_t

height\_sp(const vector<size\_t>&bins){

const auto BIN\_HEIGHT = 30;

size\_t height;

if(!(bins.size() == 0)){

size\_t max\_count = bins[0]; //поиск max из корзин

for (size\_t bin : bins)

if (bin > max\_count)

max\_count = bin;

height = max\_count\* BIN\_HEIGHT +50;

}

return height;

}

void

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

const auto IMAGE\_WIDTH = 400;

const auto TEXT\_LEFT = 20;

const auto TEXT\_BASELINE = 20;

const auto TEXT\_WIDTH = 30;

const auto BIN\_HEIGHT = 30;

const auto BLOCK\_WIDTH = 20;

const size\_t SCREEN\_WIDTH = 80;

const size\_t MAX\_ASTERISK = SCREEN\_WIDTH - 3 - 1;

size\_t max\_count = bins[0]; //поиск max из корзин

for (size\_t bin : bins)

if (bin > max\_count)

max\_count = bin;

size\_t IMAGE\_HEIGHT = height\_sp(bins);

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

double top = 0;

for (size\_t bin : bins) {

const double bin\_height = BIN\_HEIGHT \* bin;

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

svg\_rect(TEXT\_LEFT +top, TEXT\_WIDTH, BLOCK\_WIDTH, bin\_height);

top+=BLOCK\_WIDTH;

}

svg\_end();

}

**svg.h**

#ifndef SVG\_H\_INCLUDED

#define SVG\_H\_INCLUDED

#include <vector>

#include <string>

using namespace std;

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

void svg\_end();

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

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

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

size\_t

height\_sp(const vector<size\_t>& bins);

#endif // SVG\_H\_INCLUDED

**test.cpp**

#include "histogram.h"

#include "svg.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\_one(){

double min=0;

double max =0;

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

assert( min == 3);

assert (max == 3);

}

void

test\_otric() {

double min = 0;

double max = 0;

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

assert(min == -3);

assert(max == -1);

}

void

test\_equal() {

double min = 0;

double max = 0;

find\_minmax({1, 1, 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);

}

///Тесты инд. задания

void

test\_img\_empty(){

double height = 0;

height\_sp({});

assert( height == 0);

}

void

test\_img\_one(){

size\_t h = 0;

h = height\_sp({1});

assert(h == 80);

}

int

main() {

test\_positive();

test\_one();

test\_otric();

test\_equal();

test\_empty();

test\_img\_empty();

test\_img\_one();

}