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 ПО систем управления»

# «Декомпозиция программы»

# Выполнил студент группы А-01-19

# Оглов В.А.

# Проверили

# Мохов А. С

# Козлюк Д. А

Москва 2020

**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:



# Часть 3. Модульное тестирование

Написать модульный тест для функции поиска минимума и максимума.

**Индивидуальное задание**

**Вариант 13**

Отображайте гистограмму вертикально с подписями снизу.

\*

\*

\* \*

\* \*

\* \* \*

\_ \_ \_

3 5 1

Предусмотреть расчет IMAGE\_HEIGHT таким образом, чтобы вся гистограмма вмещалась в область рисунка.

**Пояснение к программе:**

Для начала необходимо найти максимальную высоту столбца, чтобы сделать это я написал функцию *max\_height.* Потом по формуле: IMAGE\_HEIGHT=(maximum\*BLOCK\_WIDTH)+30 , где maximum – найденная ранее высота, а “30” – высота, сложенная из высоты текста под гистограммой и небольшого зазора, мы находим такую высоту рисунка, чтобы туда в любом случае вмещалась вся гистограмма. Чтобы поменять вид гистограммы с горизонтального на вертикальный нужно в вызове функции svg\_rect(TEXT\_WIDTH, top-bin\_width, BIN\_HEIGHT, bin\_width ,"red","#aab5ff")

поменять ширину и высоту столбцов местами и подкорректировать координату “y” так, чтобы столбцы “росли” вверх.

**Текст программы:**

**Main.cpp**

#include <iostream>

#include <vector>

using namespace std;

#include "svg.h"

#include "histogram.h"

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

{

double min;

double max;

find\_minmax(numbers,min,max);

vector<size\_t> bins(bin\_count,0);

for (double number : numbers)

{

size\_t bin;

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

if (bin == bin\_count)

{

bin--;

}

bins[bin]++;

}

return bins;

}

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;

}

}

for (size\_t bin : bins)

{

if (bin < 100)

{

cout << ' ';

}

if (bin < 10)

{

cout << ' ';

}

cout << bin << "|";

size\_t height = bin;

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

{

cout << '\*';

}

cout << '\n';

}

}

int main()

{

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;

double min, max;

find\_minmax(numbers, min, max);

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

show\_histogram\_svg(bins);

return 0;

}

**Svg.cpp**

#include <iostream>

#include <vector>

#include "svg.h"

using namespace std;

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

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

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

}

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 max\_height(const vector<size\_t>& bins,double& maximum)

{maximum=bins[0];

for (size\_t bin : bins)

{ if (bin > maximum) {

maximum = bin;

}

}

}

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 BIN\_HEIGHT = 30;

const auto BLOCK\_WIDTH = 10;

double maximum;

max\_height(bins,maximum);

size\_t IMAGE\_HEIGHT=(maximum\*BLOCK\_WIDTH)+30;

double top = IMAGE\_HEIGHT-20;

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

double TEXT\_WIDTH=50;

for (size\_t bin : bins)

{

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

svg\_text(TEXT\_WIDTH+(BIN\_HEIGHT/4.0), top + TEXT\_BASELINE, to\_string(bin));

svg\_rect(TEXT\_WIDTH, top-bin\_width, BIN\_HEIGHT, bin\_width ,"red","#aab5ff");

TEXT\_WIDTH+=BIN\_HEIGHT;

}

svg\_end();

}

**Histogram.cpp**

#include <iostream>

#include <vector>

using namespace std;

#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;

}

}

}

}

**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);

#endif // HISTOGRAM\_H\_INCLUDED

**Svg.h**

#ifndef SVG\_H\_INCLUDED

#define SVG\_H\_INCLUDED

#include <vector>

using namespace std;

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

void

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

void

svg\_begin(double width, double height);

void

svg\_end();

void

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

#endif // SVG\_H\_INCLUDED

Также были добавлены тесты для проверки функции **max\_height**, которая участвует в индивидуальном задании:

#include "svg.h"

#include <cassert>

#include <string.h>

void test\_1()

{double maximum = 0;

max\_height({1, 2, 3}, maximum);

assert(maximum == 3);

}void

test\_same() {

double maximum = 0;

max\_height({3, 3, 3}, maximum);

assert(maximum == 3);

}

int

main() {

test\_1();

test\_same();

}

**Ссылка на репозитарий:**

[https://github.com/OglovVA/lab03](https://vk.com/away.php?to=https%3A%2F%2Fgithub.com%2FOglovVA%2Flab03&cc_key=)