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

# Макаров К.О.

# Проверили

# Мохов А. С

# Козлюк Д. А

Москва 2020

# Задание:

* 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.

#### **Вариант 9**

# Результат выполнения нужно оформить отдельным коммитом на GitHub. Доработку следует делать с использованием функции, для который нужно добавить unit-тесты из не менее двух существенно отличающихся случаев.

Запросите у пользователя ширину одного «блока» гистограммы BLOCK\_WIDTH. Не позволяйте вводить ширину блока менее 3px и более 30px. В этом случае предлагайте пользователю ввести ее заново с указанием причины. #include "svg.h"

**Описание логики решения индивидуального задания:**

Ввод ширины блока запишем как отдельный модуль, чтобы его было возможно подключить как к основной программе, так и к тестирующей программе. Чтобы задать ширину блока воспользуемся циклом while, в начале зададим flag=1, и будем совершать тело цикла пока flag не станет 0.flag станет равным 0, когда выполнится условие (BLOCK\_WIDTH>=3 && BLOCK\_WIDTH<=30). Если же выполнится условие (BLOCK\_WIDTH<3 || BLOCK\_WIDTH>30) , то программа выведет причину по которой не подходит ширина блока.

**Ссылка на репозитарий:**

[**https://github.com/MakarovKO/lab3.3.git**](https://github.com/MakarovKO/lab3.3.git)

**Код программы:**

***Main.cpp***

#include <iostream>

#include <vector>

#include <conio.h>

#include "histogram.h"

#include "svg.h"

using namespace std;

vector<double>

input\_numbers (size\_t count)

{

vector <double> result(count);

for (int i=0; i<count; i++)

{

cin>>result[i];

}

return result;

}

vector<size\_t>

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

{

vector<size\_t>bins(bin\_count,0); /\* массив из индексов корзин \*/

for(double x : numbers) /\* мы каждому x присваиваем последовательно каждый элемент массива "numbers" \*/

{

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

if (bin\_index==bin\_count) /\* индекс корзины "bin\_index"\*/

{

bin\_index=bin\_index-1; /\* нумирация идет с 0\*/

}

bins[bin\_index]++;

}

return bins;

}

int main()

{

size\_t number\_count, bin\_count;

double max=0, min=0;

cerr<<"number\_count=";

cin>>number\_count;

cerr<<"Enter number count: "<<"\n";

const auto numbers = input\_numbers(number\_count);/\* массив из количества элементов "number\_count"\*/

cerr<<"bin\_count=";

cin>> bin\_count;

find\_minmax (numbers,max,min);

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

show\_histogram\_svg(bins);

return 0;

}

***Svg.h***

#ifndef SVG\_H\_INCLUDED

#define SVG\_H\_INCLUDED

#include <vector>

using namespace std;

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

#endif // SVG\_H\_INCLUDED

***Svg.cpp***

#include "svg.h"

#include "block\_width.h"

#include <iostream>

using namespace std;

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='red' fill='blue'/>";

}

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

{

double BLOCK\_WIDTH;

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;

input\_BLOCK\_WIDTH(BLOCK\_WIDTH,1);

size\_t MAX\_ASTERISK=(size\_t)(350/BLOCK\_WIDTH);

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

unsigned max\_count=0;

for (size\_t b : bins) /\* мы присваиваем значение количества элементов \*/

{

if(max\_count<b)

max\_count=b; /\* здесь мы ищем максимальное количество элементов в массиве\*/

}

double top = 0;

for (size\_t bin : bins)

{

size\_t height=35;

if(max\_count>MAX\_ASTERISK) /\* Если количество будет больше 35, то уменьшаем масштаб\*/

height=MAX\_ASTERISK\*((static\_cast<double>(bin))/max\_count); /\* это для того,чтобы было дробное число , получаем, используя "static\_cast<double>"\*/

else

{

height=bin;

}

height = BLOCK\_WIDTH \* height;

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

svg\_rect(TEXT\_WIDTH, top, height, BIN\_HEIGHT, "red", "#aaffaa");

top += BIN\_HEIGHT;

}

svg\_end();

}

***Histogram.h***

#ifndef HISTOGRAM\_H\_INCLUDED

#define HISTOGRAM\_H\_INCLUDED

#include <vector>

using namespace std;

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

#endif // HISTOGRAM\_H\_INCLUDED

***Histogram.cpp***

#include "histogram.h"

using namespace std;

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

{

min = numbers[0];

max = numbers[0];

for(double x: numbers)

{

if(min>x)

min=x;

if(max<x)

max=x;

}

}

***Block\_width.h***

#ifndef BLOCK\_WIDTH\_H\_INCLUDED

#define BLOCK\_WIDTH\_H\_INCLUDED

#include <vector>

using namespace std;

#include <iostream>

void input\_BLOCK\_WIDTH(double &BLOCK\_WIDTH,int flag);

#endif // BLOCK\_WIDTH\_H\_INCLUDED

***Block\_width.cpp***

#include "block\_width.h"

#include <iostream>

using namespace std;

void input\_BLOCK\_WIDTH(double &BLOCK\_WIDTH, int flag)

{

while (flag==1)

{

cerr<<"Please enter the width of one block of the histogram\n";

cin>>BLOCK\_WIDTH;

if (BLOCK\_WIDTH<3 || BLOCK\_WIDTH>30)

cerr << "The width of one block of the histogram cannot be more then 30 px or less then 3 px\n";

if (BLOCK\_WIDTH>=3 && BLOCK\_WIDTH<=30)

flag=0;

}

}

***Test.cpp***

#include "histogram.h"

#include "block\_width.h"

#include <cassert>

void test\_less3()

{

double BLOCK\_WIDTH=2;

int flag=1;

input\_BLOCK\_WIDTH(BLOCK\_WIDTH, flag);

assert(flag == 1);

}

void test\_more30()

{

double BLOCK\_WIDTH=40;

int flag=1;

input\_BLOCK\_WIDTH(BLOCK\_WIDTH, flag);

assert(flag == 1);

}

void test\_positive()

{

double min = 0;

double max = 0;

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

assert(min == 1);

assert(max == 3);

}

void test\_1()

{

double min = 0;

double max = 0;

find\_minmax({-1, -7, -10}, max, min);

assert(min == -10);

assert(max == -1);

}

void test\_2()

{

double min = 0;

double max = 0;

find\_minmax({6, 6, 6}, max, min);

assert(min == 6);

assert(max == 6);

}

void test\_3()

{

double min = 0;

double max = 0;

find\_minmax({6}, max, min);

assert(min == 6);

assert(max == 6);

}

void test\_4()

{

double min = 0;

double max = 0;

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

assert(min == 0);

assert(max == 0);

}

int main()

{

test\_positive();

test\_1();

test\_2();

test\_3();

test\_4();

test\_less3();

test\_more30();

}