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 № 4 (ч.1 и ч.2)**

**По курсу «Разработка ПО систем управления»**

**«Библиотеки. Низкоуровневые операции»**

Выполнил студент группы А-01-19

Оглов В.А.

Проверили

Мохов А. С

Козлюк Д. А

Москва 2020

**Цель работы**

**1-я часть:**

1. Уметь устанавливать и подключать к программе внешние библиотеки.
2. Уметь использовать типовые элементы API: функции обратного вызова, битовые флаги и маски, массивы и строки C.
3. Уметь работать с параметрами командной строки программы.

*Задание:*

Добавить возможность построения гистограммы по данным из файла из сети. Адрес файла задается аргументом командной строки программы. Если адрес не задан, читать данные со стандартного ввода, как раньше.

Работу нужно вести на основе кода общего задания к ЛР № 3 в старом репозитарии в отдельной ветке без создания нового проекта. По этой причине во всех примерах используется lab03.exe.

Пример строки запуска:

lab03.exe http://uii.mpei.ru/study/courses/cs/lab03/marks.txt >marks.svg

**2-я часть:**

1. Уметь применять побитовые операции для типовых сценариев.
2. Уметь работать с API, принимающими указатели, в том числе строки C.
3. Знать характерные особенности документации на API библиотек.

*Задание:*

Код, который будет написан в этой части работы, должен в итоге формировать строку, которую нужно отобразить снизу итогового SVG в формате:

Windows v5.1 (build 1234)

Computer name: My-Comp

**4.1 Общее задание**

Адаптировать индивидуальное задание предыдущих ЛР к коду после рефакторинга при необходимости. Если требовалось ввести дополнительные данные, их нужно добавить в Input.

**4.2 Индивидуальные задания**

Всю необходимую информацию по функциям cURL нужно изучить по официальной документации и включить в отчет ссылку на конкретную страницу. В частности, если не указан конкретный параметр curl\_easy\_setopt(), нужно подобрать его по описанию в списке.

Во всех вариантах нужно обрабатывать ошибки cURL, как это делается в основной программе.

Если вариант требует ввода дополнительных данных, проверить работу с файлом «из сети» можно через локальный файл и адрес вида file://C:/Users/kozlyuk/Desktop/marks.txt для файла marks.txt на рабочем столе пользователя kozlyuk. Полный путь к файлу можно получить, щелкнув в адресной строке «Проводника» (так же, как перед запуском cmd).

#### **Вариант 13**

Добавьте программе опцию -stroke для задания цвета границ столбцов. Например, lab03.exe -stroke red или lab04.exe -stroke "#ff0000" делает их красными. Опция может стоять до или после URL: lab03.exe -stroke red http://... или lab04.exe http://... -fill red. Если после -stroke не стоит еще одного аргумента, нужно печатать сообщение с подсказкой, как запускать программу, и завершать работу.

**Пояснение к программе:**

Для выполнения индивидуального задания я добавил в определение функции “show\_histogram\_svg”, также сделал передачу значения переменной “stroke” в вызове функции “svg\_rect(TEXT\_WIDTH, top-bin\_width, BIN\_HEIGHT, bin\_width ,**stroke**,"#aab5ff")”. Также добавлена опция “-stroke” и учтены все частные случаи (для понимания добавлены комментарии в коде).

**Текст программы:**

**Main.cpp**

#include <curl/curl.h>

#include <iostream>

#include <vector>

#include <string>

#include <sstream>

#include "histogram.h"

#include "svg.h"

#include <windows.h>

#include <string.h>

#include <sstream>

using namespace std;

struct Options

{ string stroke;

bool stroke\_correct;

bool guide;

char\* url;};

Options parse\_args(int argc, char\*\* argv)

{Options opt;

opt.url=0;

opt.stroke\_correct=false;

opt.guide = false;

for (int i = 1; i < argc; i++) //i равно 1 тк argv[0] всегда название программы

{ if (argv[i][0] == '-')

{ //if (strcmp(argv[i],"-stroke") == 0)

if(string(argv[i]) == "-stroke")

{if(i+1<argc)

{ opt.stroke = string(argv[i+1]);

if (opt.stroke.size()) // если у аргумента после -stroke есть длина то true и значит оно введено

{opt.stroke\_correct=true;

i++; //переход к следующему аргументу }

else

{ opt.guide=true; //если введено не правильно то false }

} else //проверка на то есть ли после stroke цвет

{ opt.guide = true; } } }

else

{

opt.url=argv[i];

}

}

return opt;

}

vector<double> input\_numbers(istream& in, const size\_t count) {

vector<double> result(count);

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

in >> result[i];

}return result;

}

Input

read\_input(istream& in, bool prompt) {

Input data;

size\_t number\_count;

if (prompt)

{ cerr << "Enter number count: ";

in >> number\_count;

cerr << "Enter numbers: ";

data.numbers = input\_numbers(in, number\_count);

cerr << "Enter column count: ";

in >> data.bin\_count; }

else

{

in >> number\_count;

data.numbers = input\_numbers(in, number\_count);

in >> data.bin\_count;

} return data;

}

size\_t

write\_data(void\* items, size\_t item\_size, size\_t item\_count, void\* ctx) {

const size\_t data\_size = item\_size \* item\_count;

const char\* new\_items = reinterpret\_cast<const char\*>(items);

stringstream\* buffer = reinterpret\_cast<stringstream\*>(ctx);

buffer->write(new\_items, data\_size);

return data\_size;

}

//изменено определение функции(добавлен 2й формальный параметр)

Input

download(const string& address,const Options &opt) {

stringstream buffer;

curl\_global\_init(CURL\_GLOBAL\_ALL);

CURL \*curl = curl\_easy\_init();

if(curl) {

CURLcode res;

curl\_easy\_setopt(curl, CURLOPT\_URL, address.c\_str());

curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, write\_data);

curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, &buffer);

res = curl\_easy\_perform(curl);

if (res != CURLE\_OK)

{

cout << curl\_easy\_strerror(res) << endl;

exit(1);

}

curl\_easy\_cleanup(curl);

}

return read\_input(buffer, false);

}

string make\_info\_text()

{

stringstream buffer;

DWORD info = GetVersion();

DWORD mask = 0x0000ffff;

DWORD build;

DWORD platform = info >> 16;

DWORD version = info & mask;

DWORD version\_major = version & 0xff;

DWORD version\_minor = version >> 8;

//printf("M\_version10 = %lu\n",version\_major);

//printf("M\_version16 = %08lx\n",version\_major);

//printf("m\_version10 = %lu\n",version\_minor);

//printf("m\_version16 = %08lx\n",version\_minor);

if ((info & 0x80000000) == 0)

{

build = platform;

}

else printf("minor\_bit = %u",1);

//printf("Windows v%lu.%lu (build %lu)\n",version\_major,version\_minor,build);

char system\_name[MAX\_COMPUTERNAME\_LENGTH + 1];

DWORD Size = sizeof(system\_name);

GetComputerNameA(system\_name, &Size);

//printf("System name: %s\n", system\_name);

buffer << "Windows v" << version\_major << "." << version\_minor << " (build " << build << ")" << " " << "Computer name: " << system\_name;

return buffer.str();

}

int main(int argc, char\* argv[]) {

string info = make\_info\_text();

Input input;

Options opt=parse\_args(argc,argv);

if (opt.guide)

{

cerr<<"Error";

return 1;

}

if (opt.url)

{

input = download(opt.url,opt);

}

else

{

input = read\_input(cin, true);

}

/\* if (argc > 1)

{

input = download(argv[1]);

}

else

{

input = read\_input(cin, true);

}\*/

const auto bins = make\_histogram(input);

show\_histogram\_svg(bins,opt.stroke);

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;

} } }

// изменил определение функции, добавил string stroke, передача в svg\_rect

void

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

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 ,stroke,"#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;

} } } }

vector<size\_t> make\_histogram(const Input& data) {

vector<size\_t> result(data.bin\_count);

double min;

double max;

find\_minmax(data.numbers, min, max);

for (double number : data.numbers) {

size\_t bin = (size\_t)((number - min) / (max - min) \* data.bin\_count);

if (bin == data.bin\_count) {

bin--;

}

result[bin]++;

} return result; }

**Histogram.h**

#ifndef HISTOGRAM\_H\_INCLUDED

#define HISTOGRAM\_H\_INCLUDED

#include <vector>

using namespace std;

struct Input {

vector<double> numbers;

size\_t bin\_count; };

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

vector<size\_t> make\_histogram(const Input& data);

#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, string stroke); //добавил string stroke в определение функции

#endif // SVG\_H\_INCLUDED

**Ссылка на репозитарий:**

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