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**

**По курсу «Разработка программного обеспечения систем управления»**

**ч. 1. Библиотеки**

# ч. 2. Низкоуровневые операции

Выполнил студент группы А-02-20

Минаев Дмитрий Алексеевич

Проверил

Мохов А.С.

Козлюк Д.А.

Василькова П.Д.

Москва 2021

**ч. 1. Библиотеки**

# Цель работы

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.

**1. Сформировал структуру и прописал ввод данных:**

struct Input

{

size\_t number\_count;

vector<double> numbers;

size\_t bin\_count;

};

vector<double> input\_numbers(istream& in, 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;

if (prompt) cerr << "Enter number count: ";

size\_t number\_count;

in >> number\_count;

if (prompt) cerr << "Enter numbers: ";

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

if (prompt) cerr << "Enter bin count: ";

in >> data.bin\_count;

return data;

}

**2. Переписал функции и main():**

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

{

const auto bin\_count = data.bin\_count;

const auto numbers = data.numbers;

double min, max;

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

find\_minmax(numbers, min, max);

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

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

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;

}

int

main() {

const **auto** input = read\_input(cin, **true**);

const **auto** bins = make\_histogram(input);

show\_histogram\_svg(bins);

}

**3. Установил библиотеку cURL и реализовал её в программе (работа с буфером):**

#include <curl/curl.h>

Input download(const string& address)

{

stringstream buffer;

char \*ip;

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\_WRITEDATA, &buffer);

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

res = curl\_easy\_perform(curl);

if (res != CURLE\_OK)

{

cout << address<<endl;

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

exit(1);

}

if((res == CURLE\_OK) && !curl\_easy\_getinfo(curl, CURLINFO\_PRIMARY\_IP, &ip) && ip)

{

cerr << "IP: " << ip <<"\n";

}

curl\_easy\_cleanup(curl);

}

return read\_input(buffer, false);

}

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

{

int int\_shkal, j;

cout << "Vvedite interval shkali (ot 2 do 9):";

cin >> int\_shkal;

if (int\_shkal<2 || int\_shkal>9) //Проверка соответствия интервала шкалы по отношению к условию (от 2 до 9)

{

cout << endl << "ERROR" << endl;

}

else //Если введённый интервал соответствует условию, программа продолжает работу

{

Input input;

if (argc > 1)

{

input = download(argv[1]);

}

else

{

input = read\_input(cin, true);

}

const auto bins = make\_histogram(input);

const auto max\_name = show\_histogram\_text(bins);

shkala(max\_name, int\_shkal, j);

}

return 0;

}

**4. Реализовал код сохранения данных из сети в буфер:**

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

{

auto data\_size = item\_size \* item\_count;

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

buffer->write(reinterpret\_cast<char\*>(items), data\_size);

return data\_size;

}

**ч. 2. Низкоуровневые операции**

# Цель работы

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.

**1. Реализовал вывод версии Windows:**

string make\_info\_text()

{

stringstream buffer;

DWORD WINAPI info = GetVersion();

DWORD mask = 0x0000ffff;

DWORD platform = info >> 16;

DWORD version = info & mask;

DWORD mask2 = 0x00ff;

DWORD version\_major = version >> 8;

DWORD version\_minor = version & mask2;

DWORD build;

if ((info & 0x10000000) == 0)

{

build = platform;

}

cout << "\n\nWindows v"<<version\_minor<<"."<<version\_major<<" (build "<<build <<")\n";

**2. Реализовал получение имени ПК:**

char length[MAX\_COMPUTERNAME\_LENGTH+1];

DWORD size;

size=sizeof(length);

GetComputerName(length,&size);

cout << "Computer name: "<<length<<"\n";

return buffer.str();

}