Федеральное государственное бюджетное образовательное учреждение высшего образования   
"Национальный исследовательский университет "МЭИ"

РАЗРАБОТКА ПРОГРАММНОГО ОБСЕПЕЧЕНИЯ СИСТЕМ УПРАВЛЕНИЯ

ЛАБОРАТОРНАЯ РАБОТА №3  
«Декомпозиция и контроль корректности программ»

Студент:  
Никифорова Екатерина Владимировна  
Группа А-03-19

Преподаватель:   
Мохов Андрей Сергеевич

Москва 2020

Вариант 13

Постановка индивидуального задания

После вывода гистограммы запрашивайте у пользователя, доволен ли он результатом. Если ответ отрицательный, позвольте ввести новое количество столбцов и перестройте гистограмму. Процесс может повторяться сколько угодно раз.

Логика решения

Для отражения гистограммы вертикально нужно ввести новые переменные:

* max\_height – максимальная высота столбца
* bin\_height – высота корзины
* top – координата по y, с которой начинается столбец корзины
* bottom – координата по x, с которой начинается столбец

Все переменные будут менятся, кроме max\_height.

Нужно изменить некоторые аргументы в функциях svg\_text и svg\_rect и поменять их порядок выполнения: сначала выполняется svg\_rect, затем – svg\_text.

1) Исходный код модуля main.cpp

#include <iostream>

#include <vector>

#include "histogram.h"

#include "svg\_histogram.h"

using namespace std;

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, double max)

{

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

for (double number : numbers)

{

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

if (bin == bin\_count)

{

bin--;

}

bins[bin]++;

}

return(bins);

}

void show\_histogram\_text(vector<size\_t> bins, size\_t number\_count)

{

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;

}

}

const bool scaling\_needed = max\_count > MAX\_ASTERISK;

for (size\_t bin : bins)

{

if (bin < 100)

{

cout << ' ';

}

if (bin < 10)

{

cout << ' ';

}

cout << bin << "|";

size\_t height = bin;

if (scaling\_needed)

{

const double scaling\_factor = (double)MAX\_ASTERISK / max\_count;

height = (size\_t)(bin \* scaling\_factor);

}

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

{

cout << '\*';

}

cout << '\n';

}

}

Int main()

{

// Ввод данных

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;

// Обработка данных

double min;

double max;

find\_minmax(numbers, min, max);

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

// Вывод данных

show\_histogram\_svg(bins);

return 0;

}

2) Исходный код модуля 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

3) Исходный код модуля histogram.cpp

#include "histogram.h"

#include <iostream>

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

{

if(numbers.size() != 0)

{

min = numbers[0];

max = numbers[0];

for (double number : numbers)

{

if (number < min)

{

min = number;

}

if (number > max)

{

max = number;

}

}

}

}

4) Исходный код модуля svg\_histogram.h

#ifndef SVG\_HISTOGRAM\_H\_INCLUDED

#define SVG\_HISTOGRAM\_H\_INCLUDED

#include <iostream>

#include <vector>

using namespace std;

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

void svg\_end();

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

#endif // SVG\_HISTOGRAM\_H\_INCLUDED

5) Исходный код модуля svg\_histogram.cpp

#include "svg\_histogram.h"

#include <iostream>

#include <vector>

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\_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 << "' />";

}

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

{

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

}

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

{

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;

const auto BLOCK\_WIDTH = 10;

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

//svg\_text(TEXT\_LEFT, TEXT\_BASELINE, to\_string(bins[0]));

//svg\_rect(TEXT\_WIDTH, 0, bins[0] \* BLOCK\_WIDTH, BIN\_HEIGHT);

double top = 0;

string stroke = "black";

string fill = "green";

for (size\_t bin : bins)

{

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

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

svg\_rect(TEXT\_WIDTH, top, bin\_width, BIN\_HEIGHT, "ыblack", "green");

top += BIN\_HEIGHT;

}

svg\_end();

}

Ссылка на репозитарий

https://github.com/NikiforovaEV/cs-lab03