20.04.20

Зозуля.А.В

А-03-19

Отчёт по лабараторной работе

№1

III-вариант

Задача

Измените высоту изображения IMAGE\_HEIGHT = 700. Дайте пользователю возможность задавать высоту столбца гистограммы. Если итоговая высота гистограммы больше IMAGE\_HEIGHT, рассчитывать высоту столбца как (IMAGE\_HEIGHT / bins\_count).

Описание логики решения

Добавил возможность ввода кастамной высоты столбца через ввод непосредственно в саму функцию show\_histogram\_svg.

Чтобы узнать итоговую высоту, нужно умножить количество корзин на высоту единичного экземпляра, если она > 700 , то значение единичного столба скейлится, так чтобы уместиться, без обрезания картинки.

Исходный код

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#include <iostream>

#include <vector>

#include "histogram.h"

using namespace std;

vector<double> input(size\_t number\_count){

vector<double> numbers(number\_count);

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

cin >> numbers[i];

}

return numbers;

}

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

// Обработка данных

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

double min;

double max;

find\_minmax(numbers,min,max);

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 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 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 show\_histogram\_svg(const vector<size\_t>& bins,double BIN\_HEIGHT,double bin\_count) {

const auto IMAGE\_WIDTH = 400;

const auto IMAGE\_HEIGHT = 700;

const auto TEXT\_LEFT = 20;

const auto TEXT\_BASELINE = 20;

const auto TEXT\_WIDTH = 50;

const auto BLOCK\_WIDTH = 10;

const auto STROKE = "aqua";

const auto FILL = "darkblue";

if(bin\_count\*BIN\_HEIGHT>IMAGE\_HEIGHT){

BIN\_HEIGHT =IMAGE\_HEIGHT/bin\_count;

}

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

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

double top = 0;

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,STROKE,FILL);

top += BIN\_HEIGHT;

}

svg\_end();

}

int main() {

// Ввод данных

size\_t number\_count;

cin >> number\_count;

vector <double> numbers = input(number\_count);

size\_t bin\_count;

cin >> bin\_count;

// Height

double bin\_height;

cin >> bin\_height;

// Обработка данных

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

// Вывод данных

show\_histogram\_svg(bins,bin\_height,bin\_count);

return 0;

}

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_