РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

ОТЧЕТ

по лабораторной работе № 1

дисциплина: Компьютерная графика

Студент: Косолапов Степан Эдуардович

Группа: НПИбд-01-20

**МОСКВА**

2022 г.

**Ход работы:**

Ссылка на github с кодом: https://github.com/stepaKosolapov/computerGraphics/tree/master/lab1

Лабораторную работу будем выполнять на языке JavaScript с использованием canvasApi из HTML5 в качестве графического api.

Все отрисовки будем выполнять через функцию line, которая принимает координаты двух точек и рисует между ними прямую:

line([x0, y0], [x1, y1]) {

this.ctx.beginPath();

this.ctx.moveTo(x0, y0);

this.ctx.lineTo(x1, y1);

this.ctx.closePath();

this.ctx.stroke();

}

Все остальные функции не будут использовать canvasApi для отрисовки элементов.

Для наглядности создадим HTML документ, где и будет находиться наш canvas:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<script defer src="./index.js"></script>

<title>Лабораторная работа 1</title>

<style>

\* {

box-sizing: border-box;

}

body {

display: flex;

flex-direction: column;

align-items: center;

gap: 24px;

font-size: 20px;

height: 100vh;

}

button {

height: 40px;

width: 100px;

}

input {

height: 40px;

width: 100px;

font-size: 20px;

}

canvas {

border: 1px solid black;

background-color: azure;

}

.controls {

display: flex;

flex-direction: column;

align-items: flex-end;

gap: 8px;

}

.inputs {

display: flex;

flex-direction: column;

gap: 8px;

}

.input {

display: flex;

flex-direction: row;

justify-content: space-between;

align-items: center;

gap: 16px;

width: 450px;

}

</style>

</head>

<body onload="init()">

<div class="controls">

<div class="inputs">

<div class="input">

<label for="n">n - глубина рекурсии</label>

<input id="n" type="number" value="5"/>

</div>

<div class="input">

<label for="x0">x0 - координата х точки привязки</label>

<input id="x0" type="number" value="800"/>

</div>

<div class="input">

<label for="y0">y0 - координата y точки привязки</label>

<input id="y0" type="number" value="100"/>

</div>

<div class="input">

<label for="a">a - длина основания</label>

<input id="a" type="number" value="100"/>

</div>

<div class="input">

<label for="fi">fi - угол наклона основания</label>

<input id="fi" type="number" value="0"/>

</div>

<div class="input">

<label for="alpha">alpha - угол наклона крыши</label>

<input id="alpha" type="number" value="30"/>

</div>

</div>

<button onclick="draw()" style="width: 100px; height: 30px">Draw</button>

</div>

<canvas id="canvas" width="1500" height="600"></canvas>

</body>

</html>

Основной код расположен в файле index.js. В нем определяется класс Drawer c какими-то базовыми функциями для отрисовки, в том числе и метод line:

class Drawer {

constructor(canvas, inputs) {

this.inputs = inputs;

this.canvas = canvas;

this.ctx = canvas.getContext('2d');

this.initCanvas();

}

initCanvas() {

this.ctx.fillStyle = "black";

this.ctx.setTransform(1,0,0,-1,0,canvas.height);

}

start() {

this.clear();

const args = this.inputs.map(input => Number(input.value));

this.draw(...args);

}

draw() {}

clear() {

this.ctx.clearRect(0, 0, this.canvas.width, this.canvas.height);

}

line([x0, y0], [x1, y1]) {

this.ctx.beginPath();

this.ctx.moveTo(x0, y0);

this.ctx.lineTo(x1, y1);

this.ctx.closePath();

this.ctx.stroke();

}

rotateX({ radians, pivot, point }) {

const [ x0, y0 ] = pivot;

const [ x, y ] = point;

return (x - x0) \* Math.cos(radians) - (y - y0) \* Math.sin(radians) + x0;

}

rotateY({ radians, pivot, point }) {

const [ x0, y0 ] = pivot;

const [ x, y ] = point;

return (x - x0) \* Math.sin(radians) + (y - y0) \* Math.cos(radians) + y0;

}

rotatePoint({ radians, pivot, point }) {

return [

this.rotateX({ radians, pivot, point }),

this.rotateY({ radians, pivot, point }),

];

}

}

Еще тут присутствуют функции rotateY, rotateX и rotatePoint. Они позволяют повернуть на количество радиан = radians, точку point относительно pivot.

Основная логика содержится в классе Pifagor, который наследуется от класса Drawer. В нем есть функция pifagor, удовлетворяющая всем требованиям задачи.

class Pifagor extends Drawer {

draw() {

this.pifagor(...arguments);

}

pifagor(n, x0, y0, a, fi, alpha) {

if (n <= 0) {

return;

}

const antiAlpha = 90 - alpha;

const fiRadians = fi \* Math.PI / 180;

const alphaRadians = alpha \* Math.PI / 180;

const pivot = [ x0, y0 ];

const points = {

A: [ x0, y0 ],

B: [ x0, y0 + a ],

C: [ x0 + a, y0 + a ],

D: [ x0 + a, y0 ],

};

const leftLeg = Math.cos(alphaRadians) \* a;

const rightLeg = Math.sin(alphaRadians) \* a;

console.log(leftLeg);

console.log(rightLeg)

points.E = this.rotatePoint({

radians: alphaRadians,

pivot: points.B,

point: [ points.B[0] + leftLeg, points.B[1] ]

});

for (const pointName of Object.keys(points)) {

const point = points[pointName];

points[pointName] = this.rotatePoint({

radians: fiRadians,

pivot,

point,

});

}

console.log(points)

this.line(points.A, points.B);

this.line(points.B, points.C);

this.line(points.C, points.D);

this.line(points.D, points.A);

this.line(points.B, points.E);

this.line(points.E, points.C);

const bigHouseArgs = [

n - 1, // n

points.B[0], // x0

points.B[1], // y0

leftLeg, // a

fi + alpha, // fi

alpha, // alpha

];

const littleHouseArgs = [

n - 1, // n

points.E[0], // x0

points.E[1], // y0

rightLeg, // a

fi - antiAlpha, // fi

alpha, // alpha

];

this.pifagor(...bigHouseArgs);

this.pifagor(...littleHouseArgs);

}

}

Функция init срабатывает при загрузке странички и инициализирует объект класса Pifagor

function init() {

const canvas = document.getElementById('canvas');

const nInput = document.getElementById('n');

const x0Input = document.getElementById('x0');

const y0Input = document.getElementById('y0');

const aInput = document.getElementById('a');

const fiInput = document.getElementById('fi');

const alphaInput = document.getElementById('alpha');

const drawer = new Pifagor(

canvas,

[

nInput,

x0Input,

y0Input,

aInput,

fiInput,

alphaInput,

],

);

window.draw = () => drawer.start();

}

**Результат работы:**

**A picture containing diagram

Description automatically generated**

**A picture containing diagram

Description automatically generated**

**Diagram

Description automatically generated**

**Diagram

Description automatically generated**

**Diagram

Description automatically generated with medium confidence**

**Diagram

Description automatically generated**