БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

ИИТ БГУИР

Программное обеспечение информационных технологий

Отчет

По лабораторной работе №1

по курсу «Алгоритмы компьютерной графики»

Вариант 12

Выполнил студент группы 481064 Сорока А.А.

Проверил преподаватель: Коренская И.Н.

Минск 2016 г

**Цель работы**:

**Задание:**

Разработать программное обеспечение, обеспечивающее создание на экране семейства фигур заданной формы. Программа должна предусматривать два режима отображения графической информации:

* отображение на экране всего множества заданных фигур;
* отображение одной текущей фигуры.

Переход от первого режима ко второму может быть осуществлен за счет того, что каждая фигура прорисовывается на экране дважды:

* первый раз фигура прорисовывается заданным цветом;
* второй раз та же фигура прорисовывается с использованием цвета фона.

Для нормального восприятия фигуры следующая прорисовка объекта в обоих режимах должна выполняться с задержкой. Выбор величины задержки осуществляется студентом.

Исходные данные:

* Фигура – шестиугольник;
* Способ перемещения – хаотическое движение изменяющегося многоугольника;
* Способ изменения – уменьшение масштаба на каждом шаге на величину 1/30 ее текущего размера;
* Величина задержки – 300 миллисекунд.

**Код программы:**

index.html

<html>

<head>

<script src="script\animation.js"></script>

</head>

<body>

<div style="position:relative; margin-top:5px">

X: <input type="number" id="speedX" value="50" min=-500 max=500/>

Y: <input type="number" id="speedY" value="50" min=-500 max=500/>

<button id="startButton" onclick="startAnimation()"></button>

</div>

<canvas style="border:1px solid #ccc;" id=canvas> </canvas>

</body>

</html>

animation.js

var ctx, speedX, speedY, x, y, H, W, currentScale, trails, useTrail, animate;

window.onload = function () {

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

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

W = canvas.width = window.innerWidth - 25;

H = canvas.height = window.innerHeight - 50;

reset();

};

function reset() {

x = 590;

y = 300;

document.getElementById("startButton").textContent = "Start";

currentScale = 1;

trails = [{\_x:x, \_y:y, \_s:currentScale}];

draw(x, y, currentScale);

}

function draw(currentX, currentY, scale) {

currentScale \*= scale;

drawTriangle(currentX, currentY, currentScale, "rgba(27, 44, 245, 0.92)");

}

function drawTriangle(currentX, currentY, scale, color) {

ctx.beginPath();

ctx.fillStyle = color;

ctx.moveTo(currentX + 80 \* scale, currentY);

ctx.lineTo(currentX + 160 \* scale, currentY + 70 \* scale);

ctx.lineTo(currentX + 160 \* scale, currentY + 160 \* scale);

ctx.lineTo(currentX + 80 \* scale, currentY + 230 \* scale);

ctx.lineTo(currentX, currentY + 160 \* scale);

ctx.lineTo(currentX, currentY + 70 \* scale);

ctx.fill();

}

function startAnimation() {

if (animate) {

document.getElementById("startButton").textContent = "Start";

clearTimeout(animate);

animate = null;

}

else {

reset();

animate = true;

document.getElementById("startButton").textContent = "Stop";

speedX = document.getElementById("speedX").value;

speedY = document.getElementById("speedY").value;

useTrail = false;

animation();

}

}

function animation() {

var n = 0;

var currentX = x;

var currentY = y;

do {

n = speedX \* (Math.random() \* 4 - 2);

}

while (x + n <= 25 || x + n >= W - 160);

x += n;

do {

n = speedY \* (Math.random() \* 4 - 2);

}

while (y + n <= 95 || y + n >= H - 75);

y += n;

var stepX = (x - currentX) / 50;

var stepY = (y - currentY) / 50;

var k = 0;

function moveToPoint() {

if (k != 50) {

k++;

currentX += stepX;

currentY += stepY;

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

draw(currentX, currentY, 0.9991);

if (animate) {

window.requestAnimationFrame(moveToPoint);

}

}

else {

animate = setTimeout(animation, 300);

}

}

moveToPoint();

}

**Скриншот выполнения программы:**

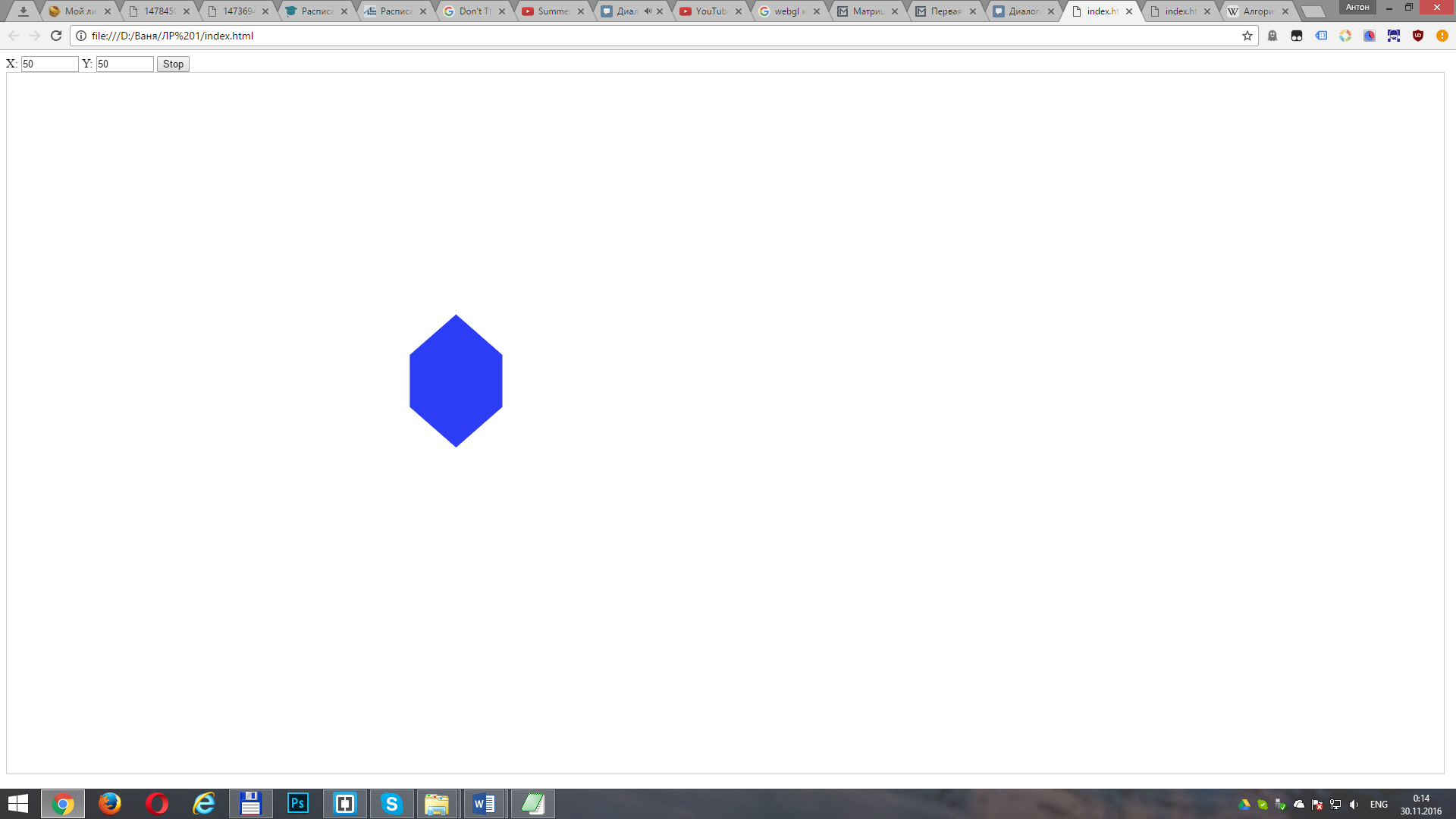


Рисунок 1 – Рабочая область приложения в ходе его работы

**Вывод**: в ходе выполнения лабораторной работы была разработано программное средство на языке JavaScript, служащее для создания анимации движущегося шестиугольника изменяющегося по ходу движения. Хаотичность движения фигуры достигается при помощи расчета новой позиции на основе введённых пользователем значений изменения скорости по осям X и Y.