**Національний технічний університет України**

**“Київський політехнічний інститут”**

**Факультет прикладної математики**

**Кафедра спеціалізованих комп’ютерних систем**

**Лабораторна робота № 3**

з дисципліни «Комп’ютерна графіка»

«Алгоритми фарбування многокутників»

Виконав:

студент групи КВ-42

**Сахнік Іван**

Перевірив:

Клятченко Я.М.

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

**Київ 2016р.**

**Завдання**

1. Зафарбувати прямокутник «легким» алгоритмом.

2. Зафарбувати багатокутник алгоритмом зі списком реберних точок.

3. Зафарбувати довільним алгоритмом багатокутник.

**Текст програми**

function Rect(x, y, width, height) {

this.width = width;

this.height = height;

this.x = x;

this.y = y;

}

Rect.prototype.draw = function() {

ctx.beginPath();

ctx.rect(this.x, this.y, this.width, this.height);

ctx.fillStyle = 'yellow';

ctx.fill();

ctx.strokeStyle = colorStroke;

ctx.stroke();

}

function Point(x, y) {

this.x = x;

this.y = y;

}

Point.prototype.draw = function(x, y) {

ctx.fillStyle = colorFill;

ctx.fillRect(x, y, 1, 1);

ctx.stroke();

}

function drawPoint(x, y, ctx) {

ctx.fillStyle = colorFill;

ctx.fillRect(x, y, 1, 1);

ctx.stroke();

}

function methodSeededFill(rect, point) {

if ((rect.x < point.x && point.x < (rect.width + rect.x)) && (rect.y < point.y && point.y < (rect.height + rect.y))) {

function drawSeeded(x, y) {

var pixelData = ctx.getImageData(x, y, 1, 1).data;

if (rgbToHex(pixelData[0], pixelData[1], pixelData[2]) != colorStroke) {

point.draw(x, y);

drawSeeded(x, y + 1);

drawSeeded(x, y - 1);

drawSeeded(x + 1, y);

drawSeeded(x - 1, y);

}

}

drawSeeded(point.x, point.y);

} else {

console.log('point don`t in rect');

}

}

function methodListOfLinePoints(ctx, pol) {

var yCur = pol[0][1];

var yMax = pol[0][1];

for (var i = 1; i < pol.length - 1; i++) {

if (pol[i][0] < yCur) {

yCur = pol[i][1];

}

if (pol[i][0] > yMax) {

yMax = pol[i][1];

}

}

while (yCur < yMax) {

var xBuf = [];

for (var i = 0; i < pol.length - 1; i++) {

if ((pol[i][1] < yCur && yCur < pol[i + 1][1]) || (pol[i][1] > yCur && yCur > pol[i + 1][1])) {

var x = (((yCur - pol[i][1]) \* (pol[i + 1][0] - pol[i][0])) / (pol[i + 1][1] - pol[i][1])) + pol[i][0];

xBuf.push(Math.ceil(x));

}

}

xBuf.sort(function(a, b) {

return a - b;

});

for (var i = 0; i < xBuf.length - 1; i += 2) {

ctx.moveTo(xBuf[i], yCur);

ctx.lineTo(xBuf[i + 1], yCur);

}

ctx.strokeStyle = colorFill;

ctx.stroke();

yCur++;

}

}

function methodXOR(ctx, pol) {

var yCur = pol[0][1];

var yMax = pol[0][1];

for (var i = 1; i < pol.length - 1; i++) {

if (pol[i][0] < yCur) {

yCur = pol[i][1];

}

if (pol[i][0] > yMax) {

yMax = pol[i][1];

}

}

while (yCur < yMax) {

var xBuf = [];

for (var i = 0; i < pol.length - 1; i++) {

if ((pol[i][1] <= yCur && yCur < pol[i + 1][1]) || (pol[i][1] > yCur && yCur >= pol[i + 1][1])) {

var x = (((yCur - pol[i][1]) \* (pol[i + 1][0] - pol[i][0])) / (pol[i + 1][1] - pol[i][1])) + pol[i][0];

xBuf.push(Math.ceil(x));

}

}

xBuf.sort(function(a, b) {

return a - b;

});

for (var i = 0; i < xBuf.length - 1; i += 2) {

for (var j = xBuf[i]; j < xBuf[i + 1]; ++j) {

drawPoint(j, yCur, ctx);

}

}

yCur++;

}

}

function componentToHex(c) {

var hex = c.toString(16);

return hex.length == 1 ? "0" + hex : hex;

}

function rgbToHex(r, g, b) {

return "#" + componentToHex(r) + componentToHex(g) + componentToHex(b);

}

const colorFill = '#aaaaaa';

const colorStroke = '#000000';

var canvas = document.getElementById('fill1');

var canvas2 = document.getElementById('fill2');

var canvas3 = document.getElementById('fill3');

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

var point = new Point(110, 110);

var rect1 = new Rect(100, 100, 80, 40);

rect1.draw();

var polygon = new Array(

new Array(60, 60),

new Array(80, 100),

new Array(80, 150),

new Array(140, 200),

new Array(33, 48),

new Array(60, 60));

ctx2 = canvas2.getContext('2d');

for (var i = 0; i < polygon.length - 1; i++) {

ctx2.moveTo(polygon[i][0], polygon[i][1]);

ctx2.lineTo(polygon[i + 1][0], polygon[i + 1][1]);

}

ctx2.strokeStyle = colorStroke;

ctx2.stroke();

ctx3 = canvas3.getContext('2d');

for (var i = 0; i < polygon.length - 1; i++) {

ctx3.moveTo(polygon[i][0], polygon[i][1]);

ctx3.lineTo(polygon[i + 1][0], polygon[i + 1][1]);

}

ctx3.strokeStyle = colorStroke;

ctx3.stroke();

function fill() {

methodListOfLinePoints(ctx2, polygon);

methodXOR(ctx3, polygon);

methodSeededFill(rect1, point);

}

