Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №7

По дисциплине «Современные платформы программирования»

Выполнила:

Студентка 3 курса

Группы ПО-3

Пивчик В.Г.

Проверил:

Крощенко А.А.

Брест 2020 г.

**Цель работы:**Освоить возможности языка программирования Java в построении графических приложений.

**Вариант 9**

Постановка задачи:

Задание 1:

Создать классы Point и Line. Объявить массив из n объектов класса Point. Для объекта класса Line определить, какие из объектов Point лежат на одной стороне от прямой линии и какие на другой. Реализовать ввод данных для объекта Line и случайное задание данных для объекта Point.

Задание 2:

Реализовать построение заданного типа фрактала по варианту.

Везде, где это необходимо, предусмотреть ввод параметров, влияющих на внешний вид фрактала.

Остров Минковского.

**Ход работы**

**Текст программы:**

**Задание 1**

**Код программы**

**Main**

package com.company;

import javax.swing.\*; import java.awt.\*;

import java.util.ArrayList; import java.util.List;

import java.util.Random; import java.util.Scanner;

public class Main extends JPanel { static final int BORDER = 390;

private List<Point> points = new ArrayList<>(); private Line line;

private Random random = new Random();

public static void main(String[] args) {

JFrame frame = new JFrame("Points and Line");

frame.add(new Main(10));

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.pack();

frame.setLocationRelativeTo(null);

frame.setVisible(true);

}

public Main(int i) { setBackground(Color.BLACK); setPreferredSize(new Dimension(400, 400));

for (int j = 0; j < i; j++) {

addCircle(BORDER, BORDER);

}

addLine();

}

public void addCircle(int maxX, int maxY) {

points.add(new Point(random.nextInt(maxX), random.nextInt(maxY)));

repaint();

}

public void addLine() {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter k and b, x1 and x2: ");

this.line = new Line(

scanner.nextInt(), scanner.nextInt(),

scanner.nextInt(),

scanner.nextInt()

);

repaint();

}

@Override

protected void paintComponent(Graphics g) { super.paintComponent(g);

for (int i = 0; i < points.size(); i++) {

Point point = points.get(i);

int currentY = line.getK() \* point.getX() + line.getB(); if (currentY < point.getY()) {

point.setColor(Color.YELLOW);

} else if (currentY > point.getY()) {

point.setColor(Color.BLUE); }

point.draw(g); }

this.line.draw(g); }

}

**Line**

package com.company;

import java.awt.\*;

public class Line {

private int k, b;

private int x1, x2, y1, y2;

public Line(int k, int b, int x1, int x2) {

this.k = k;

this.b = b;

this.x1 = x1;

this.y1 = k \* x1 + b; this.x2 = x2;

this.y2 = k \* x2 + b;

}

public void draw(Graphics g) {

Graphics2D g2d = (Graphics2D) g;

g2d.setColor(Color.GRAY);

g.drawLine(x1, y1, x2, y2);

}

public int getK() {

return k;

}

public void setK(int k) {

this.k = k;

}

public int getB() {

return b;

}

public void setB(int b) {

this.b = b;

}

}

**Point**package com.company;

import java.awt.\*;

import java.awt.geom.Ellipse2D;

public class Point {

private int x, y;

private static int width = 10, height = 10;

private Color color;

public Color getColor() {

return color;

}

public void setColor(Color color) { this.color = color;

}

public Point(int x, int y) {

this.x = x;

this.y = y;

this.color = Color.GRAY;

}

public void draw(Graphics g) {

Graphics2D g2d = (Graphics2D) g;

Ellipse2D.Double point = new Ellipse2D.Double(x, y, width, height);

g2d.setColor(color);

g2d.fill(point); }

public int getX() {

return x;

}

public void setX(int x) {

this.x = x;

}

public int getY() {

return y;

}

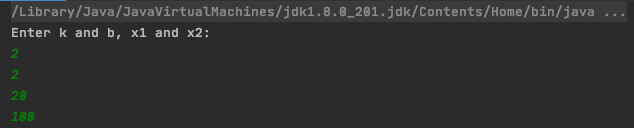
public void setY(int y) {

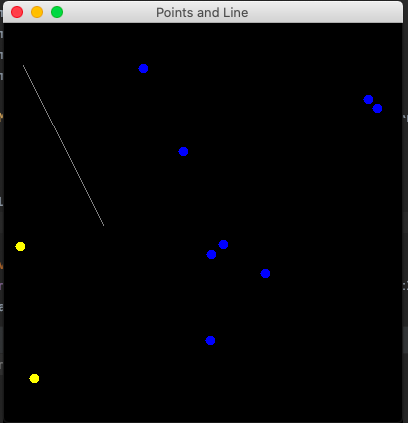
this.y = y;

}

}

**Рисунок с результатом работы программы**





**Задание 2**

**Код программы**

**Main**

package com.company;

import java.awt.\*;

import javax.swing.\*;

public class Main {

public static void main(String[] args) {

JFrame frame = new JFrame("Minkowski");

JTextField textField = new JTextField(15);

Toolkit toolkit = Toolkit.getDefaultToolkit();

Dimension screenSize = toolkit.getScreenSize();

frame.setSize(screenSize);

DrawPanel panel = new DrawPanel();

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setContentPane(panel);

frame.add(textField);

JButton button = new JButton("Send iterator");

frame.add(button);

frame.setVisible(true);

button.addActionListener(e -> {

panel.setIterations(Integer.parseInt(textField.getText()));

frame.setContentPane(panel);

});

}

}

**Line**

package com.company;

class Line {

private int x1;

private int x2;

private int y1;

private int y2;

public Line(int x1, int y1, int x2, int y2) {

this.x1 = x1;

this.x2 = x2;

this.y1 = y1;

this.y2 = y2;

}

public int getX1() {

return x1;

}

public int getX2() {

return x2;

}

public int getY1() { return y1; }

public int getY2() {

return y2;

}

}

**DrawPanel**

package com.company;

import javax.swing.\*;

import java.awt.\*;

import java.util.ArrayList;

class DrawPanel extends JPanel {

private int iterations;

public void setIterations(int iterations) {

this.iterations = iterations;

}

public static int drawMinkowski(Graphics g, int iteration, Line obj, int length) {

if (iteration == 0 || length == 0) {

g.drawLine(obj.getX1(), obj.getY1(), obj.getX2(), obj.getY2());

return 0;

}

Line arr[] = new Line[8];

//Left-right

if (obj.getY1() == obj.getY2() && obj.getX2() > obj.getX1()) {

arr[0] = new Line(obj.getX1(), obj.getY1(), obj.getX1() + length, obj.getY1());

arr[1] = new Line(obj.getX1() + length, obj.getY1(), obj.getX1() + length, obj.getY1() - length);

arr[2] = new Line(obj.getX1() + length, obj.getY1() - length,

obj.getX1() + length \* 2, obj.getY1() - length);

arr[3] = new Line(obj.getX1() + length \* 2, obj.getY1() - length, obj.getX1() + length \* 2, obj.getY1());

arr[4] = new Line(obj.getX1() + length \* 2, obj.getY1(),

obj.getX1() + length \* 2, obj.getY1() + length);

arr[5] = new Line(obj.getX1() + length \* 2, obj.getY1() + length, obj.getX1() + length \* 3, obj.getY1() + length);

arr[6] = new Line(obj.getX1() + length \* 3, obj.getY1() + length, obj.getX1() + length \* 3, obj.getY1());

arr[7] = new Line(obj.getX1() + length \* 3, obj.getY1(),

obj.getX2(), obj.getY1());

}

//Right-left

if (obj.getY1() == obj.getY2() && obj.getX2() < obj.getX1()) {

arr[0] = new Line(obj.getX1(), obj.getY1(), obj.getX1() - length, obj.getY1());

arr[1] = new Line(obj.getX1() - length, obj.getY1(), obj.getX1() - length, obj.getY1() + length);

arr[2] = new Line(obj.getX1() - length, obj.getY1() + length,

obj.getX1() - length \* 2, obj.getY1() + length);

arr[3] = new Line(obj.getX1() - length \* 2, obj.getY1() + length, obj.getX1() - length \* 2, obj.getY1());

arr[4] = new Line(obj.getX1() - length \* 2, obj.getY1(), obj.getX1() - length \* 2, obj.getY1() - length);

arr[5] = new Line(obj.getX1() - length \* 2, obj.getY1() - length, obj.getX1() - length \* 3, obj.getY1() - length);

arr[6] = new Line(obj.getX1() - length \* 3, obj.getY1() - length, obj.getX1() - length \* 3, obj.getY1());

arr[7] = new Line(obj.getX1() - length \* 3, obj.getY1(), obj.getX2(), obj.getY1());

}

//Top-down

if (obj.getX1() == obj.getX2() && obj.getY1() > obj.getY2()) {

arr[0] = new Line(obj.getX1(), obj.getY1(), obj.getX1(), obj.getY1() - length);

arr[1] = new Line(obj.getX1(), obj.getY1() - length, obj.getX1() - length, obj.getY1() - length);

arr[2] = new Line(obj.getX1() - length, obj.getY1() - length, obj.getX1() - length, obj.getY1() - length \* 2);

arr[3] = new Line(obj.getX1() - length, obj.getY1() - length \* 2, obj.getX1(), obj.getY1() - length \* 2);

arr[4] = new Line(obj.getX1(), obj.getY1() - length \* 2, obj.getX1() + length, obj.getY1() - length \* 2);

arr[5] = new Line(obj.getX1() + length, obj.getY1() - length \* 2, obj.getX1() + length, obj.getY1() - length \* 3);

arr[6] = new Line(obj.getX1() + length, obj.getY1() - length \* 3, obj.getX1(), obj.getY1() - length \* 3);

arr[7] = new Line(obj.getX1(), obj.getY1() - length \* 3, obj.getX2(), obj.getY2());

}

//Down-town

if (obj.getX1() == obj.getX2() && obj.getY1() < obj.getY2()) {

arr[0] = new Line(obj.getX1(), obj.getY1(), obj.getX1(), obj.getY1() + length);

arr[1] = new Line(obj.getX1(), obj.getY1() + length, obj.getX1() + length, obj.getY1() + length);

arr[2] = new Line(obj.getX1() + length, obj.getY1() + length, obj.getX1() + length, obj.getY1() + length \* 2);

arr[3] = new Line(obj.getX1() + length, obj.getY1() + length \* 2, obj.getX1(), obj.getY1() + length \* 2);

arr[4] = new Line(obj.getX1(), obj.getY1() + length \* 2, obj.getX1() - length, obj.getY1() + length \* 2);

arr[5] = new Line(obj.getX1() - length, obj.getY1() + length \* 2, obj.getX1() - length, obj.getY1() + length \* 3);

arr[6] = new Line(obj.getX1() - length, obj.getY1() + length \* 3, obj.getX1(), obj.getY1() + length \* 3);

arr[7] = new Line(obj.getX1(), obj.getY1() + length \* 3, obj.getX2(), obj.getY2());

}

iteration--;

length = length / 4;

for (int i = 0; i < 8; i++) {

drawMinkowski(g, iteration, arr[i], length);

}

return 0;

}

@Override

protected void paintComponent(Graphics g) {

super.paintComponent(g); this.setBackground(Color.black);

g.setColor(Color.pink);

Toolkit kit = Toolkit.getDefaultToolkit(); Dimension size = kit.getScreenSize();

//Square side

int length = 400;

//Upper left angle

int x1 = (int) size.getHeight() - length / 2 - 150;

int y1 = 250;

ArrayList<Line> arr = new ArrayList<Line>();

Line one = new Line(x1, y1, x1 + length, y1);

Line two = new Line(x1 + length, y1, x1 + length, y1 + length);

Line three = new Line(x1 + length, y1 + length, x1, y1 + length);

Line four = new Line(x1, y1 + length, x1, y1);

arr.add(one);

arr.add(two);

arr.add(three);

arr.add(four);

drawMinkowski(g, iterations, one, length / 4);

drawMinkowski(g, iterations, two, length / 4);

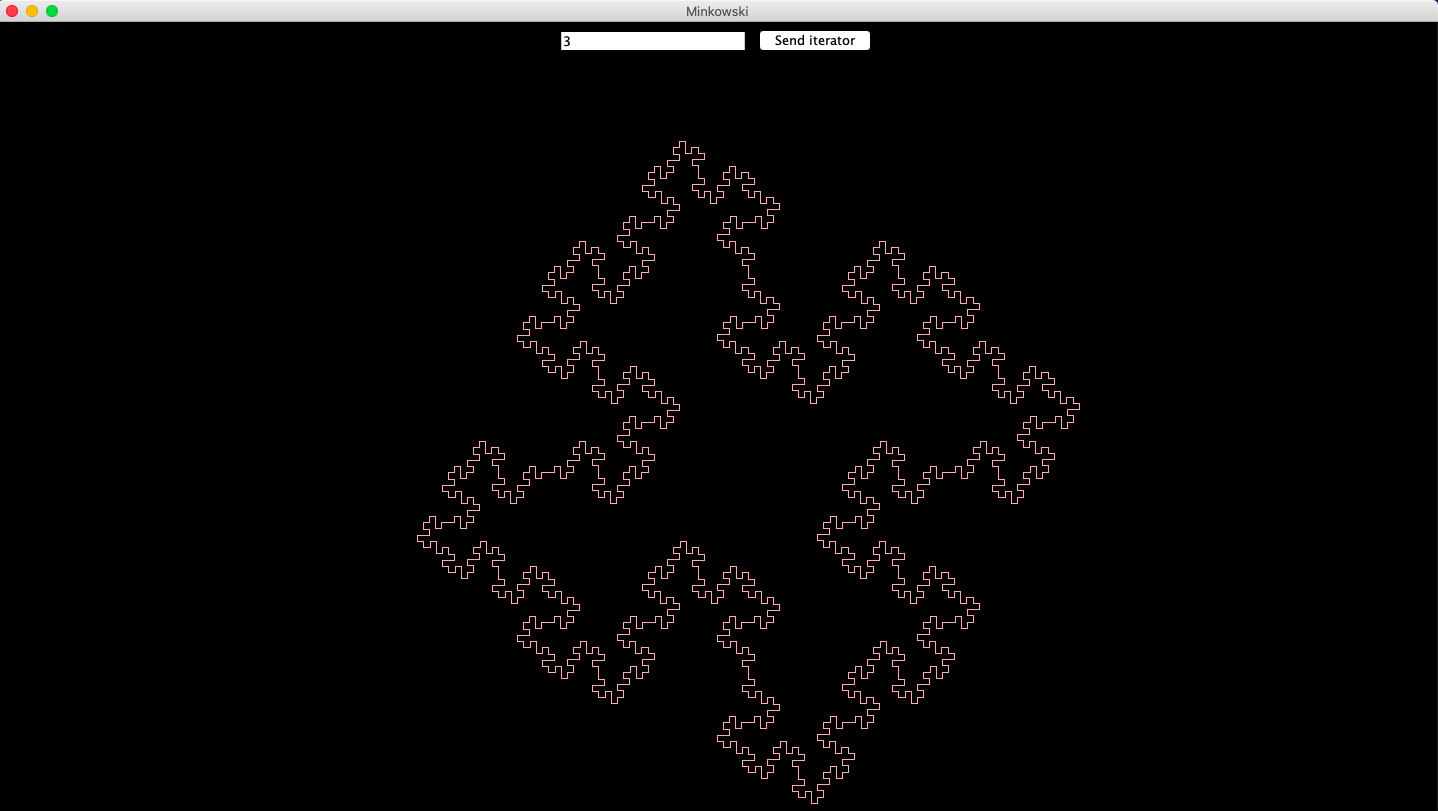
drawMinkowski(g, iterations, three, length / 4);

drawMinkowski(g, iterations, four, length / 4);

}

}

**Рисунок с результатом работы программы**

****

**Выводы**:  
Я освоила возможности языка программирования Java в построении графических приложений.