

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
Кафедра ИИТ

ОТЧЁТ

по лабораторной работе №7
за 1 семестр 3 курса

Выполнил:
студент группы ПО-9(1)
3 курса
Зейденс Никита
Вячеславович

Проверил:
Крощенко
А. А.

Брест, 2024

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

Вариант 5

Задание 1: Построение графических примитивов и надписей.

Требования к выполнению:

- Реализовать соответствующие классы, указанные в задании;
- Организовать ввод параметров для создания объектов (можно использовать файлы);
- Осуществить визуализацию графических примитивов, решить поставленную задачу.

Изобразить в окне приложения (апплета) отрезок, вращающийся в плоскости экрана вокруг одной из своих концевых точек. Цвет прямой должен изменяться при переходе от одного положения к другому.

Код программы:

```
package org.example.graphics;

import javafx.animation.Animation;
import javafx.animation.RotateTransition;
import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.scene.shape.Line;
import javafx.stage.Stage;
import javafx.util.Duration;

public class RotatingLineApp extends Application {

    private static final double LINE_LENGTH = 200;
    private static final double DURATION_SECONDS = 3;

    @Override
    public void start(Stage primaryStage) {
        TextField x1Field = new TextField();
        TextField y1Field = new TextField();
        TextField x2Field = new TextField();
        TextField y2Field = new TextField();
        TextField strokeWidthField = new TextField();

        Button startButton = new Button("Start");
        startButton.setOnAction(event -> {
            double x1 = Double.parseDouble(x1Field.getText());
            double y1 = Double.parseDouble(y1Field.getText());
            double x2 = Double.parseDouble(x2Field.getText());
            double y2 = Double.parseDouble(y2Field.getText());
            double strokeWidth =
                Double.parseDouble(strokeWidthField.getText());

            Line line = createLine(x1, y1, x2, y2, strokeWidth);
            RotateTransition rotateTransition =
                createRotationAnimation(line);
```

```

        Group root = new Group(line);
        Scene scene = new Scene(root, 400, 400);
        primaryStage.setScene(scene);
        primaryStage.setTitle("Rotating Line");
        primaryStage.show();

        rotateTransition.play();
    });

    VBox vbox = new VBox(
        new Label("Enter line coordinates and stroke width:"),
        new Label("x1: "),
        x1Field,
        new Label("y1: "),
        y1Field,
        new Label("x2: "),
        x2Field,
        new Label("y2: "),
        y2Field,
        new Label("Stroke Width: "),
        strokeWidthField,
        startButton
    );
    vbox.setSpacing(10);
    vbox.setPadding(new Insets(10));

    Scene scene = new Scene(vbox, 400, 400);
    primaryStage.setScene(scene);
    primaryStage.setTitle("Rotating Line");
    primaryStage.show();
}

private Line createLine(double x1, double y1, double x2, double y2,
double strokeWidth) {
    Line line = new Line(x1, y1, x2, y2);
    line.setStroke(Color.RED);
    line.setStrokeWidth(strokeWidth);
    return line;
}

private RotateTransition createRotationAnimation(Line line) {
    double centerX = line.getStartX();
    double centerY = line.getStartY();

    RotateTransition rotateTransition = new
RotateTransition(Duration.seconds(DURATION_SECONDS), line);
    rotateTransition.setByAngle(0);
    rotateTransition.setCycleCount(Animation.INDEFINITE);
    rotateTransition.setAutoReverse(false);

    // Change line color during animation
    rotateTransition.currentTimeProperty().addListener((observable,
oldValue, newValue) -> {
        double progress = newValue.toSeconds() / DURATION_SECONDS;
        if (progress <= 0.5) {
            line.setStroke(Color.RED.interpolate(Color.GREEN, progress *
2));
        } else {
            line.setStroke(Color.GREEN.interpolate(Color.RED, (progress -
0.5) * 2));
        }
    });

    // Calculate the new coordinates for the line based on the

```

```

rotation around the endpoint
        double angle = progress * 360;
        double newX = centerX + Math.cos(Math.toRadians(angle)) *
LINE_LENGTH;
        double newY = centerY + Math.sin(Math.toRadians(angle)) *
LINE_LENGTH;


        line.setEndX(newX);
        line.setEndY(newY);
    });

    return rotateTransition;
}

public static void main(String[] args) {
    launch(args);
}
}

```

Результат работы:

 Rotating Line
 —
□
×

Enter line coordinates and stroke width:

x1:

123

y1:

200

x2:

200


y2:

123

Stroke Width:

5

Start

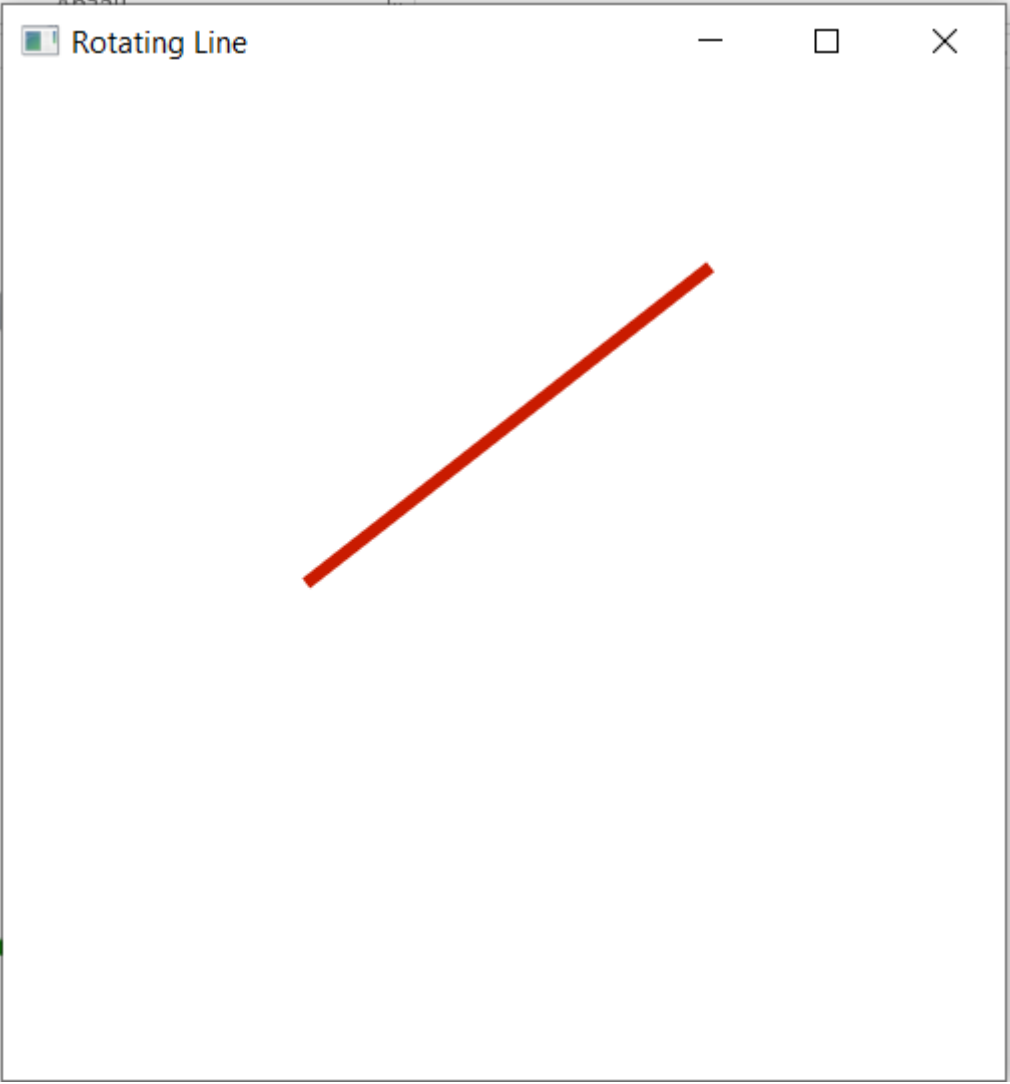
 Rotating Line

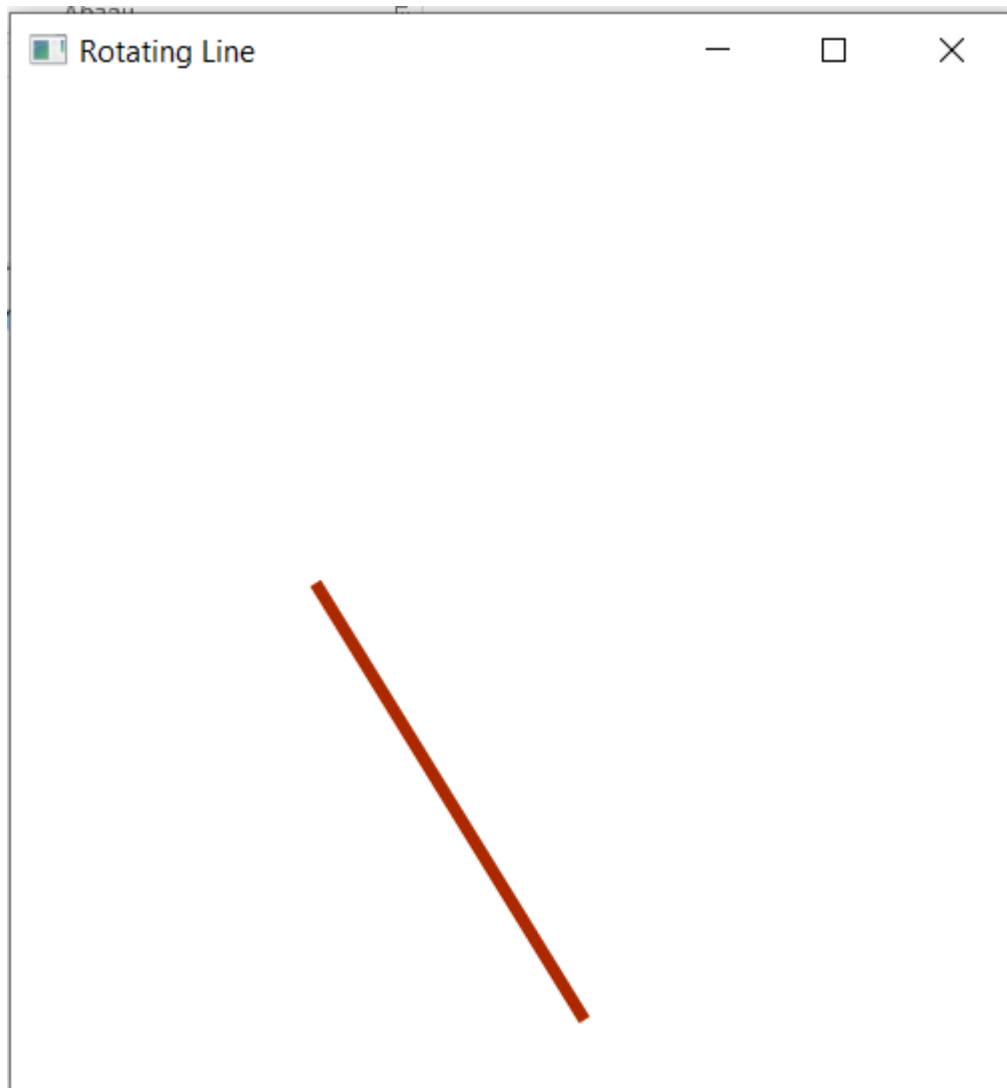
—

□

×







Задание 2: Реализовать построение заданного типа фрактала по варианту.

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

Дерево Пифагора.

Код программы:

```
package org.example.pifagor;

import javafx.application.Application;
import javafx.geometry.Insets;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.control.Button;
import javafx.scene.control.Label;
import javafx.scene.control.TextField;
import javafx.scene.layout.GridPane;
import javafx.scene.layout.HBox;
import javafx.scene.layout.VBox;
import javafx.scene.paint.Color;
import javafx.stage.Stage;
```

```

public class PythagorasTreeFractal extends Application {

    private static final int WIDTH = 800; // Ширина окна
    private static final int HEIGHT = 600; // Высота окна

    private static final double INITIAL_ANGLE = Math.PI / 2; // Начальный
угол
    private static final double INITIAL_LENGTH = 200; // Начальная длина
    private static final int MAX_DEPTH = 10; // Максимальная глубина рекурсии

    private TextField angleTextField;
    private TextField lengthTextField;
    private TextField depthTextField;
    private Canvas canvas;
    private GraphicsContext gc;

    @Override
    public void start(Stage primaryStage) {
        primaryStage.setTitle("Pythagoras Tree Fractal");

        canvas = new Canvas(WIDTH, HEIGHT);
        gc = canvas.getGraphicsContext2D();
        drawPythagorasTree(INITIAL_ANGLE, INITIAL_LENGTH, MAX_DEPTH);

        angleTextField = new TextField(String.valueOf(INITIAL_ANGLE));
        lengthTextField = new TextField(String.valueOf(INITIAL_LENGTH));
        depthTextField = new TextField(String.valueOf(MAX_DEPTH));

        Button drawButton = new Button("Нарисовать");
        drawButton.setOnAction(e -> {
            double angle = Double.parseDouble(angleTextField.getText());
            double length = Double.parseDouble(lengthTextField.getText());
            int depth = Integer.parseInt(depthTextField.getText());
            drawPythagorasTree(angle, length, depth);
        });

        GridPane settingsPane = new GridPane();
        settingsPane.setHgap(10);
        settingsPane.setVgap(10);
        settingsPane.setPadding(new Insets(10));
        settingsPane.addRow(0, new Label("Угол:"), angleTextField);
        settingsPane.addRow(1, new Label("Длина:"), lengthTextField);
        settingsPane.addRow(2, new Label("Глубина:"), depthTextField);

        HBox controlBox = new HBox(10, drawButton);
        VBox root = new VBox(10, settingsPane, controlBox, canvas);

        primaryStage.setScene(new Scene(root));
        primaryStage.show();
    }

    private void drawPythagorasTree(double angle, double length, int depth) {
        gc.clearRect(0, 0, WIDTH, HEIGHT);
        drawPythagorasTree(gc, WIDTH / 2, HEIGHT, angle, length, 0, depth);
    }

    private void drawPythagorasTree(GraphicsContext gc, double x, double y,
double angle, double length, int currentDepth, int maxDepth) {
        if (currentDepth > maxDepth) {
            return;
        }

        double x2 = x + Math.cos(angle) * length;
        double y2 = y - Math.sin(angle) * length;
    }
}

```



```

gc.setStroke(Color.BLACK);
gc.setLineWidth(1.0);
gc.strokeLine(x, y, x2, y2);

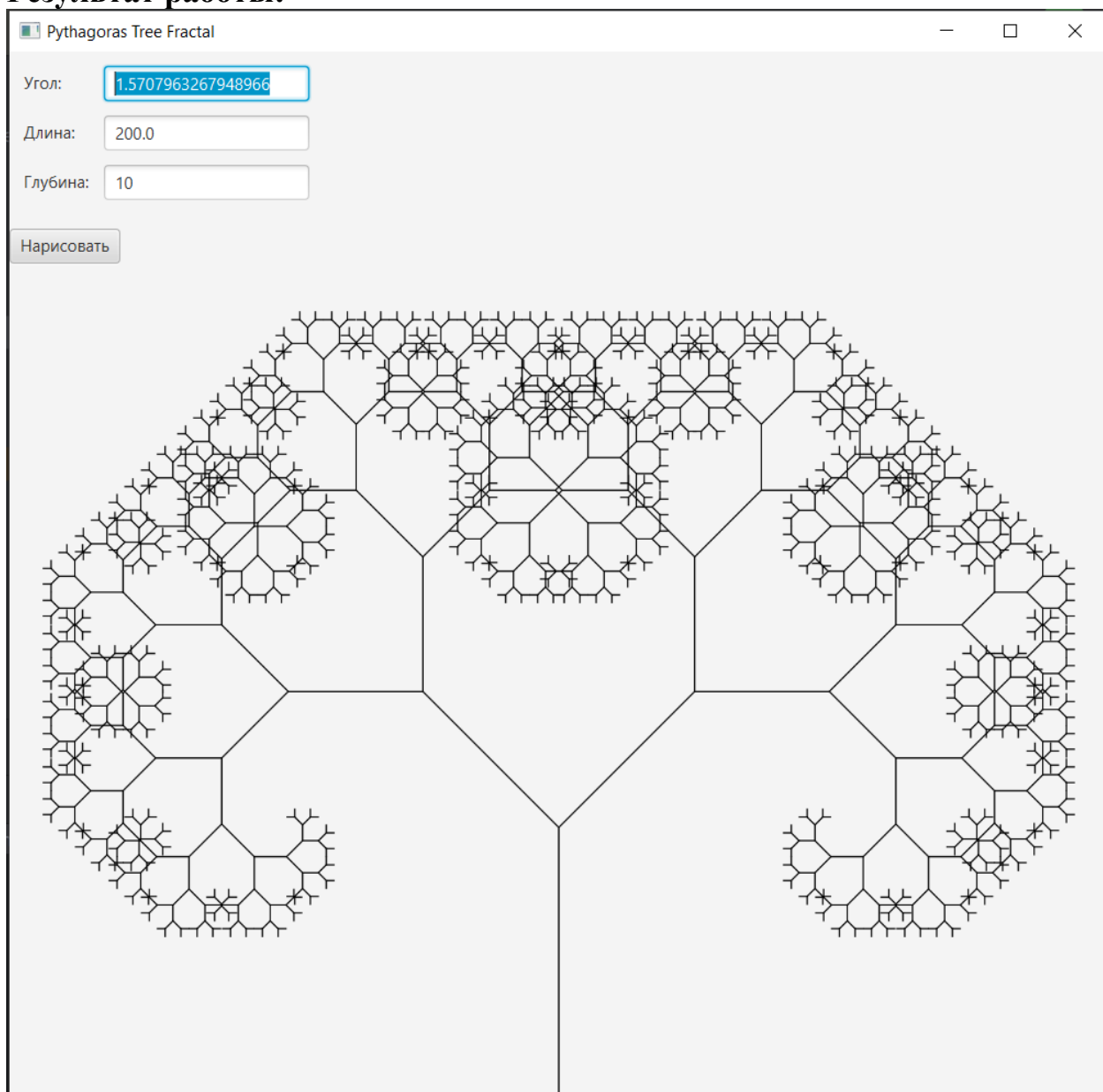
double angle1 = angle - Math.PI / 4;
double angle2 = angle + Math.PI / 4;
double lengthRatio = 0.7; // Коэффициент уменьшения длины

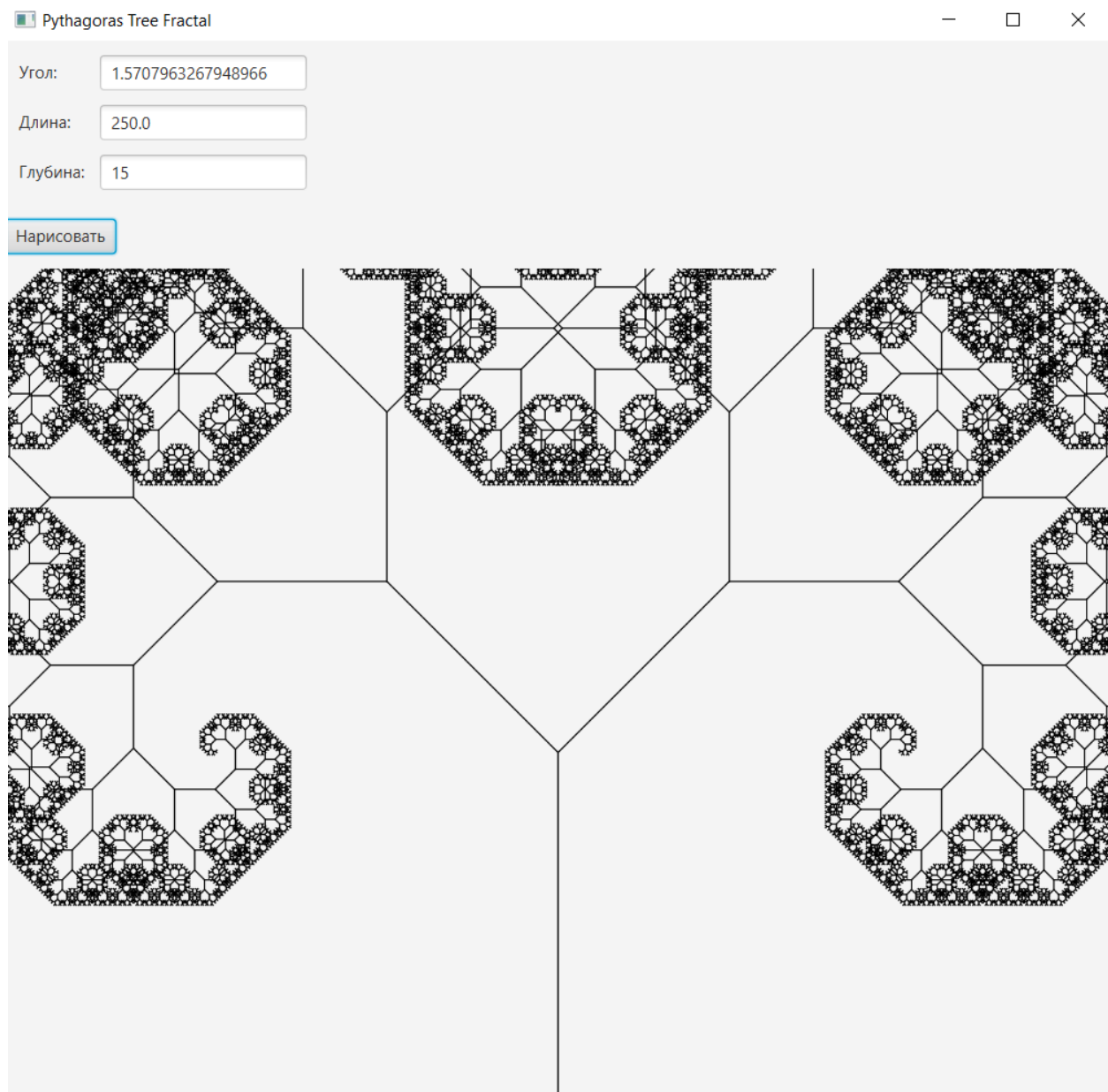
drawPythagorasTree(gc, x2, y2, angle1, length * lengthRatio,
currentDepth + 1, maxDepth);
drawPythagorasTree(gc, x2, y2, angle2, length * lengthRatio,
currentDepth + 1, maxDepth);
}

public static void main(String[] args) {
    launch(args);
}
}

```

Результат работы:





Вывод: Возможности языка программирования Java в построении графических приложений были изучены и применены на практике.