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

ОТЧЁТ

по лабораторной работе №7

Выполнила:

студентка 3 курса
группы ПО-9
Шубич Дарья
Константинова

Проверил:

Крощенко А.А.

Брест 2024

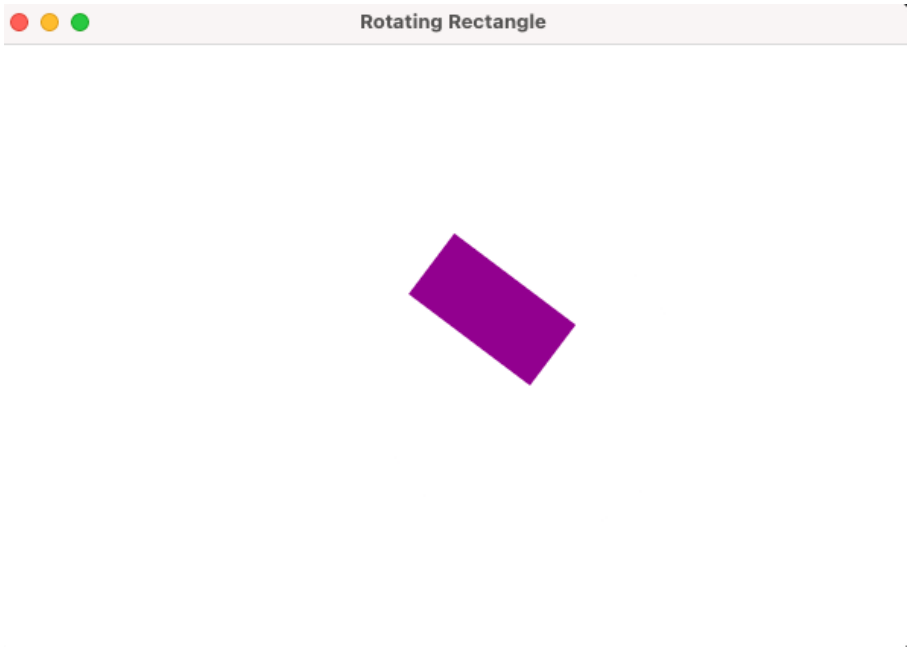
Цель работы:
освоить возможности языка программирования Java в построении графических приложений

Вариант 11

Задание 1

Изобразить прямоугольник, вращающийся в плоскости фрейма вокруг одной из своих вершин.

Результат программы



Код программы:

```
package org.example;
import javafx.animation.Animation;
import javafx.animation.KeyFrame;
import javafx.animation.Timeline;
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.paint.Color;
import javafx.scene.shape.Rectangle;
import javafx.stage.Stage;
import javafx.util.Duration;

public class RotatingRectangle extends Application {

    private static final double FRAME_WIDTH = 600;
    private static final double FRAME_HEIGHT = 400;
    private static final double RECTANGLE_WIDTH = 100;
    private static final double RECTANGLE_HEIGHT = 50;
    private static final double ROTATION_SPEED = 1;

    private Rectangle rectangle;
```

```
    private double centerX;
    private double centerY;
    private double angle = 0;

    @Override
    public void begin(Stage primaryStage) {
        rectangle = new Rectangle(RECTANGLE_WIDTH, RECTANGLE_HEIGHT,
Color.PURPLE);
        centerX = FRAME_WIDTH / 2;
        centerY = FRAME_HEIGHT / 2;
        rectangle.setX(centerX - RECTANGLE_WIDTH / 2);
        rectangle.setY(centerY - RECTANGLE_HEIGHT / 2);

        Group root = new Group(rectangle);
        Scene scene = new Scene(root, FRAME_WIDTH, FRAME_HEIGHT,
Color.WHITE);

        Timeline timeline = new Timeline(new KeyFrame(Duration.millis(16),
event -> rotateRectangle()));
        timeline.setCycleCount(Animation.INDEFINITE);
        timeline.play();

        primaryStage.setScene(scene);
        primaryStage.setTitle("Rotating Rectangle");
        primaryStage.show();
    }

    private void rotateRectangle() {
        double newX = centerX + (rectangle.getX() - centerX) *
Math.cos(Math.toRadians(ROTATION_SPEED)) - (rectangle.getY() - centerY) *
Math.sin(Math.toRadians(ROTATION_SPEED));
        double newY = centerY + (rectangle.getX() - centerX) *
Math.sin(Math.toRadians(ROTATION_SPEED)) + (rectangle.getY() - centerY) *
Math.cos(Math.toRadians(ROTATION_SPEED));

        angle += ROTATION_SPEED;

        rectangle.setX(newX);
        rectangle.setY(newY);
        rectangle.setRotate(angle);
    }

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

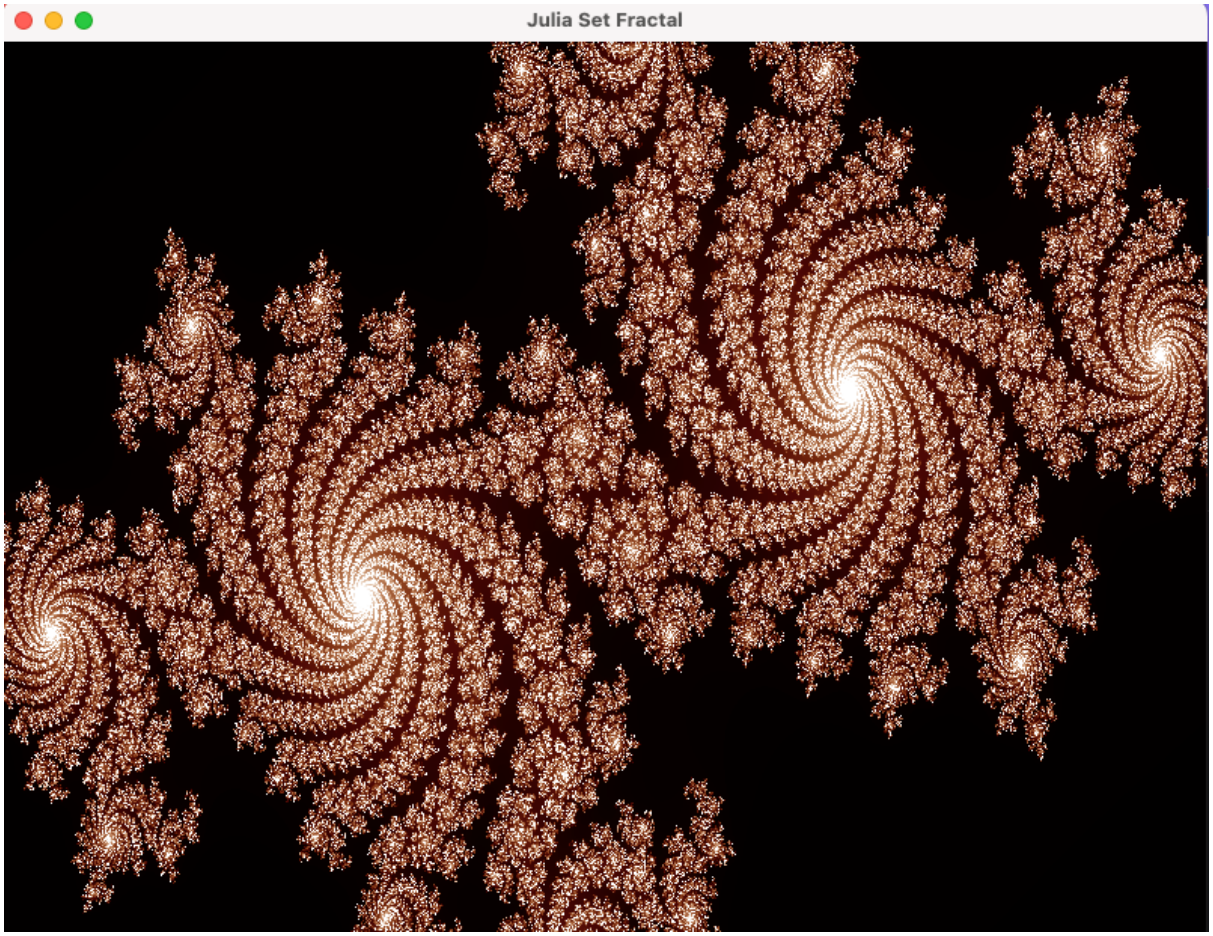
Задание 2

Реализовать построение заданного типа фрактала по варианту

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

Множество Жюлиа

Выходные данные:



Код программы:

```
package org.example;
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.image.PixelWriter;
import javafx.scene.layout.BorderPane;
import javafx.scene.paint.Color;
import javafx.stage.Stage;

public class JuliaSetFractal extends Application {

    private static final int WIDTH = 800;
    private static final int HEIGHT = 600;
    private static final int MAX_ITERATIONS = 500;
    private static final double SCALE = 500;
    private static final double OFFSET_X = WIDTH / 2;
    private static final double OFFSET_Y = HEIGHT / 2;

    private double cRe = -0.7;
    private double cIm = 0.27015;

    @Override
    public void begin(Stage primaryStage) {
```

```
        Canvas canvas = new Canvas(WIDTH, HEIGHT);
        drawJuliaSet(canvas);

        BorderPane root = new BorderPane(canvas);
        Scene scene = new Scene(root);

        primaryStage.setScene(scene);
        primaryStage.setTitle("Julia Set Fractal");
        primaryStage.show();
    }

    private void drawJuliaSet(Canvas canvas) {
        GraphicsContext gc = canvas.getGraphicsContext2D();
        PixelWriter pw = gc.getPixelWriter();

        for (int x = 0; x < WIDTH; x++) {
            for (int y = 0; y < HEIGHT; y++) {
                double zx = 1.5 * (x - OFFSET_X) / (SCALE);
                double zy = (y - OFFSET_Y) / (SCALE);
                int iteration = 0;
                while (zx * zx + zy * zy < 4 && iteration < MAX_ITERATIONS)
{
                    double temp = zx * zx - zy * zy + cRe;
                    zy = 2.0 * zx * zy + cIm;
                    zx = temp;
                    iteration++;
                }
                if (iteration == MAX_ITERATIONS) {
                    pw.setColor(x, y, Color.WHITE);
                } else {
                    double ratio = (double) iteration / MAX_ITERATIONS;
                    pw.setColor(x, y, Color.color(ratio, ratio * ratio,
ratio * ratio * ratio));
                }
            }
        }
    }

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