МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
“БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ”
КАФЕДРА ИНТЕЛЛЕКТУАЛЬНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Отчёт
по лабораторной работе №7

Выполнил:
студент группы ПО-9
Зеленков К. И.

Проверил:
Крощенко А. А.

Брест 2024

Вариант 6

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

Задание 1

Задать движение окружности по апплету так, чтобы при касании границы окружность отражалась от нее

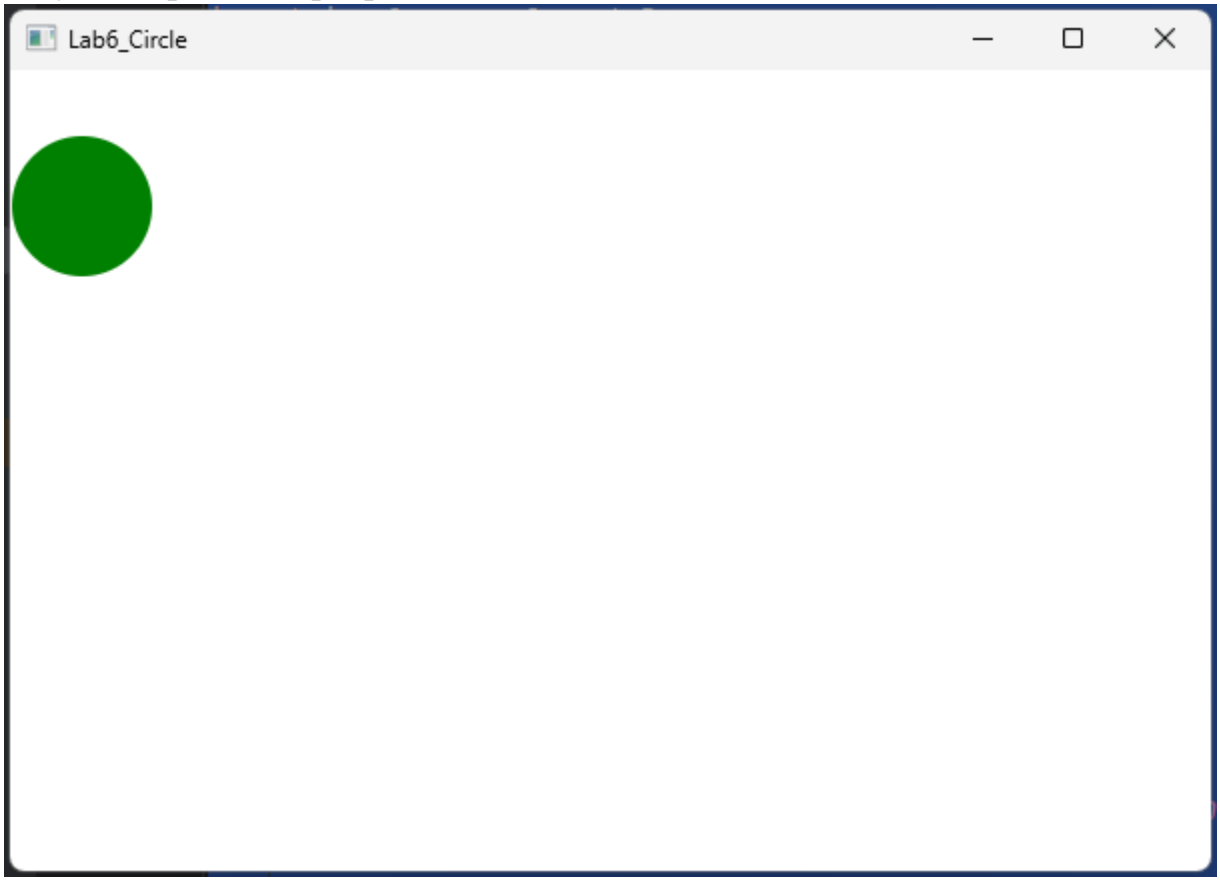
Код программы:

MovingCircle.java:

```
package ru.akimov.spring.lab7spp;

import javafx.animation.AnimationTimer;
import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.layout.Pane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Circle;
import javafx.stage.Stage;

public class MovingCircle extends Application {
    private static final int WIDTH = 600;
    private static final int HEIGHT = 400;
    private static final int CIRCLE_RADIUS = 35;
    private static final int SPEED = 2;
    private double circleX = WIDTH / 2;
    private double circleY = HEIGHT / 2;
    private double deltaX = SPEED;
    private double deltaY = SPEED;
    @Override
    public void start(Stage primaryStage) {
        Pane root = new Pane();
        Scene scene = new Scene(root, WIDTH, HEIGHT);
        Circle circle = new Circle(circleX, circleY, CIRCLE_RADIUS);
        circle.setFill(Color.GREEN);
        root.getChildren().add(circle);
        AnimationTimer timer = new AnimationTimer() {
            @Override
            public void handle(long now) {
                moveCircle();
                checkBoundaryCollision(circle);
            }
        };
        timer.start();
        primaryStage.setScene(scene);
        primaryStage.setTitle("Lab6_Circle");
        primaryStage.show();
    }
    private void moveCircle() {
        circleX += deltaX;
        circleY += deltaY;
    }
    private void checkBoundaryCollision(Circle circle) {
        if (circleX - CIRCLE_RADIUS <= 0 || circleX + CIRCLE_RADIUS >=
                WIDTH) {
            deltaX *= -1;
        }
        if (circleY - CIRCLE_RADIUS <= 0 || circleY + CIRCLE_RADIUS >=
                HEIGHT) {
```

```
            deltaY *= -1;
        }
        circle.setCenterX(circleX);
        circle.setCenterY(circleY);
    }
    public static void main(String[] args) {
        launch(args);
    }
}
```

Результат работы программы:



Задание 2

Реализовать построение заданного типа фрактала по варианту

6) Склоненное дерево Пифагора (обдуваемое ветром)

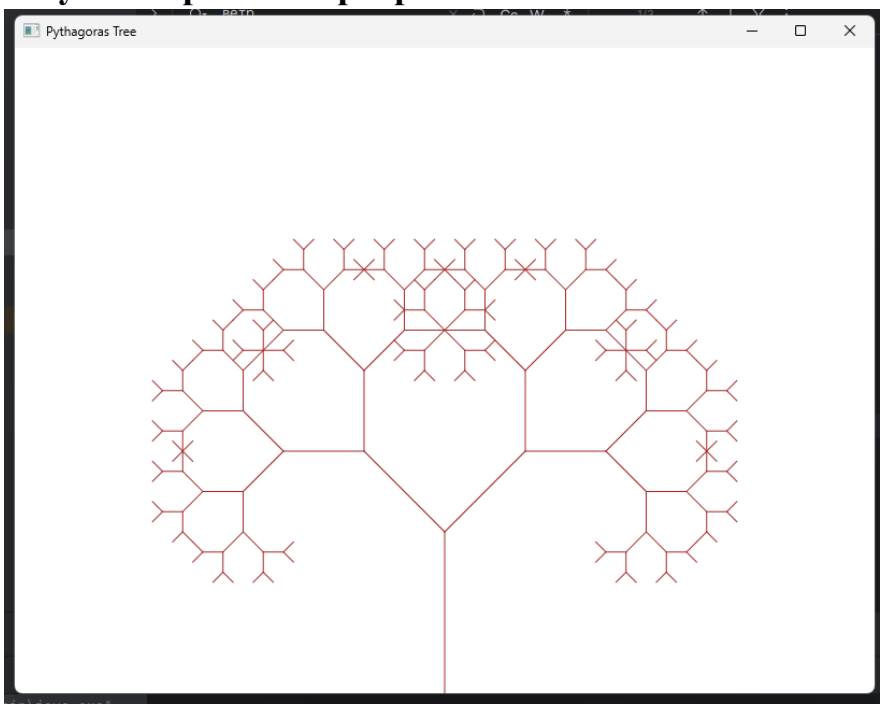
Код программы

Tree.java:

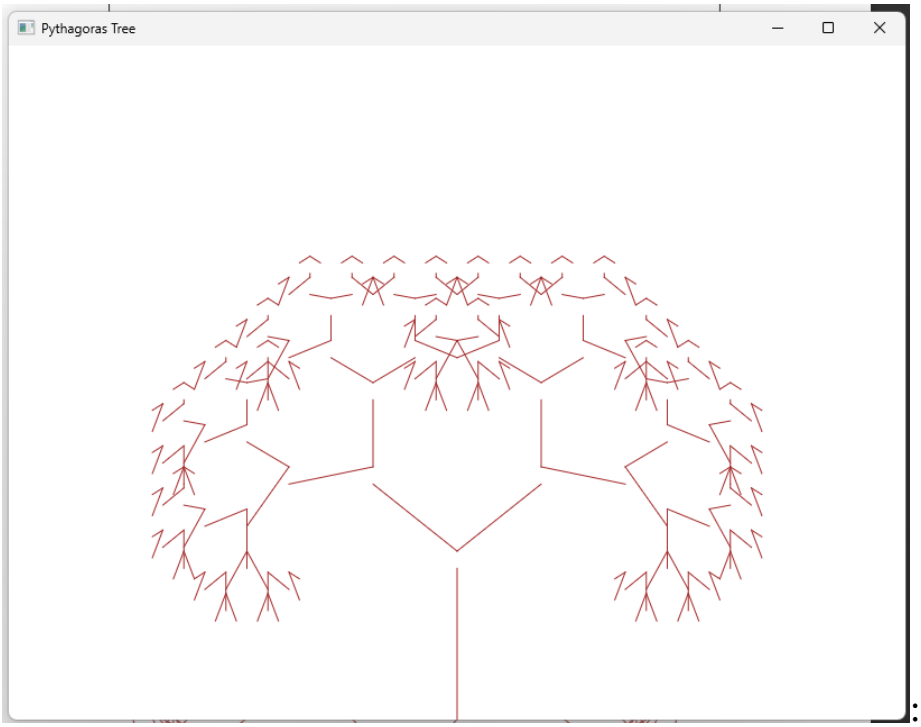
```
package ru.akimov.spring.lab7spp;

import javafx.animation.AnimationTimer;
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.canvas.Canvas;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.paint.Color;
import javafx.stage.Stage;

public class Tree extends Application {
    private static final int WIDTH = 800;
    private static final int HEIGHT = 600;
```

```
    private static final int ANGLE = 45; // Угол наклона веток
    private static final double WIND_FORCE = 0.4; // Сила ветра
    private GraphicsContext gc;
    private double windOffset = 0;

    @Override
    public void start(Stage primaryStage) {
        Canvas canvas = new Canvas(WIDTH, HEIGHT);
        gc = canvas.getGraphicsContext2D();
        primaryStage.setTitle("Pythagoras Tree");
        primaryStage.setScene(new Scene(new Group(canvas)));
        primaryStage.show();
        drawTree(WIDTH / 2, HEIGHT, HEIGHT / 4, -90, 8); // Начальные
параметры дерева
        new AnimationTimer() {
            @Override
            public void handle(long now) {
                windOffset = Math.sin(now * 1e-9) * WIND_FORCE * HEIGHT /
                        10;
                gc.clearRect(0, 0, WIDTH, HEIGHT);
                drawTree(WIDTH / 2, HEIGHT, HEIGHT / 4, -90, 10);
            }
        }.start();
    }

    private void drawTree(double x, double y, double len, double angle, int
            depth) {
        if (depth == 0) return;
        double x2 = x + Math.cos(Math.toRadians(angle)) * len;
        double y2 = y + Math.sin(Math.toRadians(angle)) * len;
        gc.setStroke(Color.BROWN);
        gc.strokeLine(x, y, x2, y2 + windOffset); // Учитываем смещение от
ветра
        drawTree(x2, y2, len * Math.sqrt(0.5), angle + ANGLE, depth - 1);
        drawTree(x2, y2, len * Math.sqrt(0.5), angle - ANGLE, depth - 1);
    }

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

Результат работы программы:



:обычное состояние



:обдуваемое ветром

Вывод: освоил возможности языка программирования Java в построении графических приложений