МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
“КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО”

Факультет прикладної математики
Кафедра програмного забезпечення комп’ютерних систем

Лабораторна робота № 3

з дисципліни “Математичні та алгоритмічні основи комп’ютерної графіки”

Виконав
студент III курсу
групи КП-83
Мричко Богдан
(*прізвище, ім’я, по батькові*)

Зарахована
“____” “_____” 20___ р.
викладачем
Шкурат Оксаною Сергіївною
(*прізвище, ім’я, по батькові*)

варіант № 15

Київ 2020

Варіант завдання

Завдання: За допомогою примітивів JavaFX максимально реально зобразити персонажа за варіантом та виконати його 2D анімацію. Для анімації скористатися стандартними засобами бібліотеки JavaFX.

Обов’язковою є реалізація таких видів анімації:

1) переміщення; 2) поворот; 3) масштабування.

Завдання за варіантом:

15	

Код програми

Main.java

```
package sample;

import javafx.animation.*;
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.paint.Color;
import javafx.scene.paint.Paint;
import javafx.stage.Stage;
import javafx.util.Duration;

public class Main extends Application {

    Paint backGroundColor = Color.rgb(120, 255, 184);

    Group root;
    Scene scene;

    @Override
    public void start(Stage primaryStage) throws Exception{
        Group root = new Group();
        Scene scene = new Scene (root, 600, 500);
        scene.setFill(backGroundColor);

        root.getChildren().add(Ground.getGround());
        root.getChildren().add(Ground.getPath());
        root.getChildren().addAll(House.getHouse());
        root.getChildren().add(Tree.getTrunk());
        root.getChildren().add(Tree.getLeaves());



        int duration = 3000;

        ScaleTransition scaleFrom = new
ScaleTransition(Duration.millis(duration), root);
        scaleFrom.setToX(1);
        scaleFrom.setToY(1);

        ScaleTransition scaleTo = new
ScaleTransition(Duration.millis(duration), root);
        scaleTo.setToX(0.5);
        scaleTo.setToY(0.5);

        RotateTransition rotate = new
RotateTransition(Duration.millis(duration), root);
        rotate.setByAngle(360f);
        rotate.setCycleCount(Timeline.INDEFINITE);

        TranslateTransition translateTo = new
TranslateTransition(Duration.millis(duration * 2), root);
        translateTo.setFromX(-100);
        translateTo.setToX(150);
        translateTo.setCycleCount(Timeline.INDEFINITE);
        translateTo.setAutoReverse(true);
```

```
        TranslateTransition translateFrom = new
TranslateTransition(Duration.millis(duration * 2), root);
        translateFrom.setFromX(150);
        translateFrom.setToX(-100);
        translateFrom.setCycleCount(Timeline.INDEFINITE);
        translateFrom.setAutoReverse(true);

        SequentialTransition scale = new SequentialTransition();
        scale.getChildren().addAll(
                scaleTo,
                scaleFrom
        );
        scale.setCycleCount(Timeline.INDEFINITE);

        SequentialTransition translate = new SequentialTransition();
        translate.getChildren().addAll(
                translateTo,
                translateFrom
        );
        translate.setCycleCount(Timeline.INDEFINITE);

        ParallelTransition animations = new ParallelTransition();
        animations.getChildren().addAll(
                scale,
                rotate,
                translate
        );
        animations.play();


        primaryStage.setTitle("House");
        primaryStage.setScene(scene);
        primaryStage.show();
    }


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

Ground.java

```
package sample;

import javafx.scene.paint.*;
import javafx.scene.shape.Ellipse;
import javafx.scene.shape.Polygon;
import javafx.scene.shape.StrokeType;

public class Ground {
    static Color groundColor = Color.rgb(81, 156, 0);
    static Paint pathColor = Color.rgb(184, 157, 0);

    static public Ellipse getGround() {
        Ellipse ground = new Ellipse();
        ground.setCenterX(300.0f);
```

```
        ground.setCenterY(250.0f);
        ground.setRadiusX(150.0f);
        ground.setRadiusY(75.0f);

        RadialGradient gradient = new RadialGradient(0,
                .1,
                450,
                300,
                150,
                false,
                CycleMethod.NO_CYCLE,
                new Stop(0, Color.YELLOW),
                new Stop(1, groundColor));

        ground.setFill(gradient);
        return ground;
    }

    static public Polygon getPath() {
        Polygon path = new Polygon(
                300.0, 250.0,
                310.0, 255.0,
                313.0, 260.0,
                310.0, 265.0,
                300.0, 280.0,
                270.0, 300.0,
                260.0, 310.0,
                250.0, 320.0,

                250.0+30, 323.0,
                260.0+30, 310.0,
                275.0+30, 300.0,
                300.0+30, 280.0,
                310.0+30, 265.0,
                313.0+30, 260.0,
                310.0+30, 255.0,
                310.0+30, 250.0,
                300.0+30, 240.0
        );
        path.setFill(pathColor);
        path.setStroke(Color.BLACK);
        path.setStrokeType(StrokeType.OUTSIDE);
        return path;
    }

}
```

House.java

```
package sample;

import javafx.scene.paint.*;
import javafx.scene.shape.Ellipse;
import javafx.scene.shape.Polygon;
import javafx.scene.shape.StrokeType;

public class House {
```

```
static Color houseDarkSideColor = Color.rgb(252, 251, 225);
static Color houseBrightSideColor = Color.rgb(230, 228, 202);

static Color houseRoofDarkSideColor = Color.rgb(219, 131, 252);
static Color houseRoofBrightSideColor = Color.rgb(241, 204, 255);

static Color houseDoorColor = Color.rgb(89, 37, 0);
static Color houseWindowColor = Color.rgb(255, 247, 23);
static double [] cornerPoint = {280.0, 260.0};

static public Polygon[] getHouse() {
    Polygon [] house  = {
            getHouseDarkSide(),
            getHouseBrightSide(),
            getHouseDoor(),
            getHouseWindow(),
            getHouseBrightRoofSide(),
            getHouseDarkRoofSide()
    };
    return house;
}

static public Polygon getHouseDarkSide() {
    Polygon wall = new Polygon(
            cornerPoint[0], cornerPoint[1],
            cornerPoint[0]+70, cornerPoint[1]-30,
            cornerPoint[0]+70, cornerPoint[1]-110,
            cornerPoint[0], cornerPoint[1]-80
    );
    wall.setFill(houseDarkSideColor);
    wall.setStroke(Color.BLACK);
    wall.setStrokeType(StrokeType.OUTSIDE);
    return wall;
}

static public Polygon getHouseBrightSide() {
    Polygon wall = new Polygon(
            cornerPoint[0], cornerPoint[1],
            cornerPoint[0]-60, cornerPoint[1]-30,
            cornerPoint[0]-60, cornerPoint[1]-110,
            cornerPoint[0], cornerPoint[1]-80
    );
    wall.setFill(houseBrightSideColor);
    wall.setStroke(Color.BLACK);
    wall.setStrokeType(StrokeType.OUTSIDE);
    return wall;
}

static public Polygon getHouseDoor() {
    Polygon door = new Polygon(
            cornerPoint[0]+20, cornerPoint[1]-10,
            cornerPoint[0]+50, cornerPoint[1]-23,
            cornerPoint[0]+50, cornerPoint[1]-83,
            cornerPoint[0]+20, cornerPoint[1]-70
    );
    door.setFill(houseDoorColor);
    door.setStroke(Color.BLACK);
    door.setStrokeType(StrokeType.OUTSIDE);
    return door;
}

static public Polygon getHouseWindow() {
    Polygon window = new Polygon(
```

```
                cornerPoint[0]-15, cornerPoint[1]-25,
                cornerPoint[0]-45, cornerPoint[1]-40,
                cornerPoint[0]-45, cornerPoint[1]-85+10,
                cornerPoint[0]-15, cornerPoint[1]-70+10
        );
        window.setFill(houseWindowColor);
        window.setStroke(Color.BLACK);
        window.setStrokeType(StrokeType.OUTSIDE);
        return window;
    }

    static public Polygon getHouseBrightRoofSide() {
        Polygon window = new Polygon(
                cornerPoint[0]-5, cornerPoint[1]-70,
                cornerPoint[0]+80, cornerPoint[1]-105,
                cornerPoint[0], cornerPoint[1]-170

        );
        window.setFill(houseRoofBrightSideColor);
        window.setStroke(Color.BLACK);
        window.setStrokeType(StrokeType.OUTSIDE);
        return window;
    }

    static public Polygon getHouseDarkRoofSide() {
        Polygon window = new Polygon(
                cornerPoint[0]-5, cornerPoint[1]-70,
                cornerPoint[0], cornerPoint[1]-170,
                cornerPoint[0]-70, cornerPoint[1]-102
        );
        window.setFill(houseRoofDarkSideColor);
        window.setStroke(Color.BLACK);
        window.setStrokeType(StrokeType.OUTSIDE);
        return window;
    }
}
```

Tree.java

```
package sample;

import javafx.scene.paint.*;
import javafx.scene.shape.Ellipse;
import javafx.scene.shape.Polygon;
import javafx.scene.shape.Rectangle;
import javafx.scene.shape.StrokeType;

public class Tree {
    static Color leavesColor = Color.rgb(67, 212, 0);
    static Paint trunkColor = Color.rgb(125, 71, 0);

    static public Rectangle getTrunk() {
        Rectangle trunk = new Rectangle();
        trunk.setX(400);
        trunk.setY(170);
        trunk.setWidth(10);
        trunk.setHeight(100);
```

```
        trunk.setFill(trunkColor);
        return trunk;
    }

    static public Ellipse getLeaves() {
        Ellipse leaves = new Ellipse();
        leaves.setCenterX(405);
        leaves.setCenterY(170);
        leaves.setRadiusX(30);
        leaves.setRadiusY(40);
        RadialGradient gradient = new RadialGradient(0,
                .1,
                380,
                180,
                70,
                false,
                CycleMethod.NO_CYCLE,
                new Stop(0, Color.GREEN),
                new Stop(1, leavesColor));

        leaves.setFill(gradient);
        return leaves;
    }
}
```

Результат