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 “Структура файлів формату .bmp. Анімація примітивів за допомогою засобів бібліотеки JavaFX”

Виконав
студент III курсу
групи КП-82
Жиров Даниїл
(*прізвище, ім’я, по батькові*)

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

варіант № 6

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.

Студентам пропонується скористатися розглянутими класами для читання, обробки та збереження зображень формату .bmp з метою використання рисунку для створення траєкторії руху або меж, в яких дозволений рух об’єктів. В даному випадку рекомендується використовувати кольори великої контрастності для різних призначень (наприклад, чорний колір відповідатиме за траєкторію руху, а інші кольори – заборонятимуть рух).

**Варіант:**

# Лістинг коду програми

```
package lab3;

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

import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.FileInputStream;
import java.io.FileOutputStream;

public class PrintingImage extends Application{

	private HeaderBitmapImage image;
	private int numberOfPixels;

	public PrintingImage() {}

	public PrintingImage(HeaderBitmapImage image) {
		this.image = image;
	}

	@Override
	public void start(Stage primaryStage) throws Exception {

		ReadingImageFromFile.loadBitmapImage("/Users/daniilzhyrov/Desktop/
JavaFX_Lab2_for_Comp_Graphics_Labs/sources/trajectory.bmp");
		this.image = ReadingImageFromFile.pr.image;
		int width = (int) this.image.getWidth();
		int height = (int) this.image.getHeight();
		int half = (int) image.getHalfOfWidth();

		Group root = new Group();
		Scene scene = new Scene(root, width, height + 200);
		scene.setFill(Color.BLACK);
//		Circle cir;

		int let = 0;
		int let1 = 0;
		int let2 = 0;
		char[][] map = new char[width][height];
		BufferedInputStream reader = new BufferedInputStream (new
FileInputStream("pixels.txt"));

		for(int i=0;i<height;i++) {
		   for(int j=0;j<half;j++) {
			let = reader.read();
			let1=let;
			let2=let;
			let1=let1&(0xf0);
			let1=let1>>4;
			let2=let2&(0x0f);
			 if(j*2<width) {
```

```
                                //cir = new Circle ((j)*2,(height-1-i),1,
Color.valueOf((returnPixelColor(let1))));
                                      //root.getChildren().add(cir);
                                       if (returnPixelColor(let1) == "BLACK")
                                    {
                                        map[j*2][height-1-i] = '1';
                                        numberOfPixels++;
                                    }
                                    else
                                    {
                                        map[j*2][height-1-i] = '0';
                                    }
                                }
                                if(j*2+1<width)
                                {
                                    //cir = new Circle ((j)*2+1,(height-1-
i),1,Color.valueOf((returnPixelColor(let2))));
                                     //root.getChildren().add(cir);
                                     if (returnPixelColor(let2) == "BLACK")
                                    {
                                        map[j*2+1][height-1-i] = '1';
                                        numberOfPixels++;
                                    }
                                    else
                                    {
                                        map[j*2+1][height-1-i] = '0';
                                    }
                                }
                            }
                        }
                        primaryStage.setScene(scene);
                    primaryStage.show();

                        reader.close();

                        int[][] black;
                        black = new int[numberOfPixels][2];
                        int lich = 0;

                        BufferedOutputStream writer = new BufferedOutputStream (new
FileOutputStream("map.txt"));
                        for(int i=0;i<height;i++) {
                           for(int j=0;j<width;j++) {
                                   if (map[j][i] == '1') {
                                          black[lich][0] = j;
                                          black[lich][1] = i;
                                          lich++;
                                   }
                                   writer.write(map[j][i]);
                           }
                           writer.write(10);
                        }
                        writer.close();

                        System.out.println("number of black color pixels = " + numberOfPixels);

                        Path path2 = new Path();
                        for (int l=0; l<numberOfPixels-1; l++)
                        {
                           path2.getElements().addAll(
                                       new MoveTo(black[l][0],black[l][1]),
```

```
                new LineTo (black[l+1][0],black[l+1][1])
    );
}

Group g = new Group();

final Circle cir = new Circle ();
cir.setCenterX(64);
cir.setCenterY(64);
cir.setRadius(64);
cir.setFill(Color.YELLOW);
g.getChildren().add(cir);

Group smile = new Group();

Arc smile1 = new Arc();
smile1.setCenterX(64.0f);
smile1.setCenterY(64.0f);
smile1.setRadiusX(48.0f);
smile1.setRadiusY(48.0f);
smile1.setStartAngle(180.0f);
smile1.setLength(180.0f);
smile1.setType(ArcType.ROUND);
smile1.setFill(Color.BLACK);
smile.getChildren().add(smile1);

Arc smile2 = new Arc();
smile2.setCenterX(64.0f);
smile2.setCenterY(64.0f);
smile2.setRadiusX(48.0f);
smile2.setRadiusY(40.0f);
smile2.setStartAngle(180.0f);
smile2.setLength(180.0f);
smile2.setType(ArcType.ROUND);
smile2.setFill(Color.WHITE);
smile.getChildren().add(smile2);

Arc smile3 = new Arc();
smile3.setCenterX(64.0f);
smile3.setCenterY(64.0f);
smile3.setRadiusX(48.0f);
smile3.setRadiusY(32.0f);
smile3.setStartAngle(179.0f);
smile3.setLength(182.0f);
smile3.setType(ArcType.ROUND);
smile3.setFill(Color.YELLOW);
smile.getChildren().add(smile3);

g.getChildren().add(smile);

Group leftEye = new Group();

Arc learc1 = new Arc();
learc1.setCenterX(40.0f);
learc1.setCenterY(55.0f);
learc1.setRadiusX(16.0f);
learc1.setRadiusY(27.0f);
learc1.setStartAngle(0.0f);
learc1.setLength(200.0f);
learc1.setType(ArcType.ROUND);
```

```
learc1.setFill(Color.WHITE);
leftEye.getChildren().add(learc1);

Arc learc2 = new Arc();
learc2.setCenterX(40.0f);
learc2.setCenterY(55.0f);
learc2.setRadiusX(12.0f);
learc2.setRadiusY(16.0f);
learc2.setStartAngle(0.0f);
learc2.setLength(200.0f);
learc2.setType(ArcType.ROUND);
learc2.setFill(Color.BLUE);
leftEye.getChildren().add(learc2);

Arc learc3 = new Arc();
learc3.setCenterX(40.0f);
learc3.setCenterY(55.0f);
learc3.setRadiusX(10.0f);
learc3.setRadiusY(12.0f);
learc3.setStartAngle(0.0f);
learc3.setLength(200.0f);
learc3.setType(ArcType.ROUND);
learc3.setFill(Color.BLACK);
leftEye.getChildren().add(learc3);

g.getChildren().add(leftEye);

Group rightEye = new Group();

Arc rearc1 = new Arc();
rearc1.setCenterX(90.0f);
rearc1.setCenterY(55.0f);
rearc1.setRadiusX(16.0f);
rearc1.setRadiusY(27.0f);
rearc1.setStartAngle(-20.0f);
rearc1.setLength(200.0f);
rearc1.setType(ArcType.ROUND);
rearc1.setFill(Color.WHITE);
rightEye.getChildren().add(rearc1);

Arc rearc2 = new Arc();
rearc2.setCenterX(90.0f);
rearc2.setCenterY(55.0f);
rearc2.setRadiusX(12.0f);
rearc2.setRadiusY(16.0f);
rearc2.setStartAngle(-20.0f);
rearc2.setLength(200.0f);
rearc2.setType(ArcType.ROUND);
rearc2.setFill(Color.BLUE);
rightEye.getChildren().add(rearc2);

Arc rearc3 = new Arc();
rearc3.setCenterX(90.0f);
rearc3.setCenterY(55.0f);
rearc3.setRadiusX(10.0f);
rearc3.setRadiusY(12.0f);
rearc3.setStartAngle(-20.0f);
rearc3.setLength(200.0f);
rearc3.setType(ArcType.ROUND);
rearc3.setFill(Color.BLACK);
```

```
        rightEye.getChildren().add(rearc3);

        g.getChildren().add(rightEye);

        Line eyebrow1 = new Line();
        eyebrow1.setStartX(27);
        eyebrow1.setStartY(27);
        eyebrow1.setEndX(50);
        eyebrow1.setEndY(24);
        eyebrow1.setStrokeWidth(3);
        g.getChildren().add(eyebrow1);

        Line eyebrow2 = new Line();
        eyebrow2.setStartX(80);
        eyebrow2.setStartY(24);
        eyebrow2.setEndX(103);
        eyebrow2.setEndY(27);
        eyebrow2.setStrokeWidth(3);

        g.getChildren().add(eyebrow2);

        root.getChildren().add(g);

        PathTransition pathTransition = new PathTransition();
        pathTransition.setDuration(Duration.millis(5000));
        pathTransition.setPath(path2);
        pathTransition.setNode(g);


        RotateTransition rotateTransition = new RotateTransition(Duration.seconds(2), g);
        rotateTransition.setByAngle(360);
        rotateTransition.setFromAngle(-180);
        rotateTransition.setCycleCount(2);

        ScaleTransition scaleTransition =
                        new ScaleTransition(Duration.seconds(4.0), g);
        scaleTransition.setToX(-2f);
        scaleTransition.setToY(-2f);


        ParallelTransition parallelTransition = new ParallelTransition();
        parallelTransition.getChildren().addAll(
                pathTransition,
                rotateTransition,
                scaleTransition
        );

        parallelTransition.play();
}

private String returnPixelColor (int color) {
        String col = "BLACK";
        switch(color) {
              case 0: return "BLACK";
              case 1: return "LIGHTCORAL";
              case 2: return "GREEN";
              case 3: return "BROWN";
              case 4: return "BLUE";
              case 5: return "MAGENTA";
              case 6: return "CYAN";
```

```
                    case 7: return "LIGHTGRAY";
                    case 8: return "DARKGRAY";
                    case 9: return "RED";
                    case 10:return "LIGHTGREEN";
                    case 11:return "YELLOW";
                    case 12:return "LIGHTBLUE";
                    case 13:return "LIGHTPINK";
                    case 14:return "LIGHTCYAN";
                    case 15:return "WHITE";
            }
            return col;
        }

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

}
```

## Результат