

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-81  Ладуди Данила Володимировича  (*прізвище, ім’я, по батькові*)  варіант № 9 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

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).

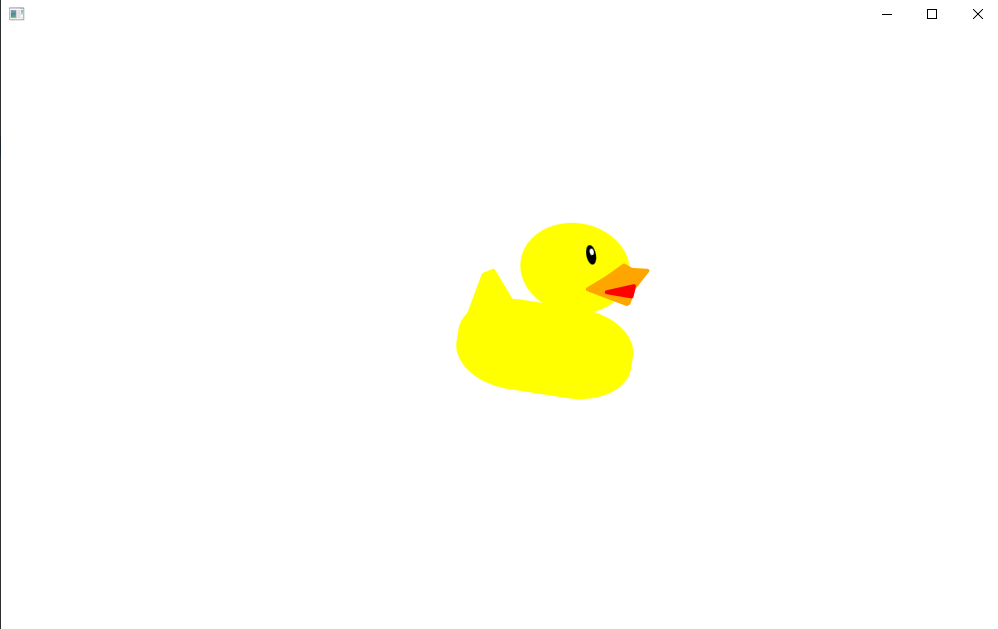
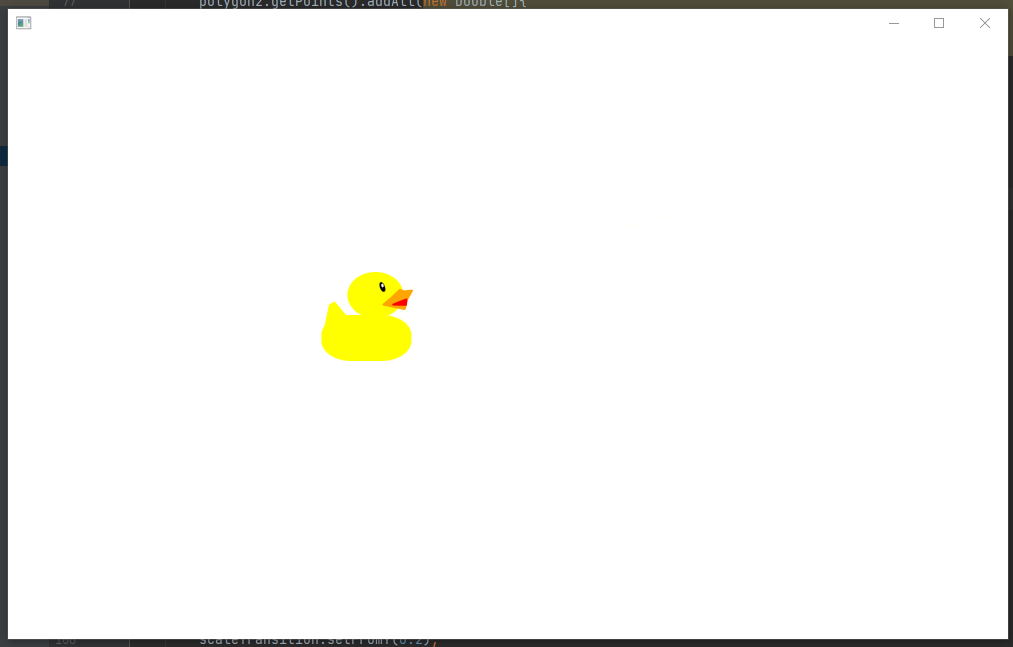
**Варіант:**



**Лістинг коду програми**

package sample;  
  
import javafx.animation.\*;  
import javafx.application.Application;  
import javafx.scene.Group;  
import javafx.scene.Scene;  
import javafx.scene.transform.Rotate;  
import javafx.stage.Stage;  
import javafx.scene.shape.\*;  
import javafx.scene.paint.Color;  
import javafx.util.Duration;  
  
public class Main extends Application {  
  
 @Override  
 public void start(Stage primaryStage) throws Exception{  
  
  
  
 Group gr = new Group();  
 primaryStage.setScene(new Scene(gr, 1000, 600));  
  
 Ellipse ellipse = new Ellipse(350,170,140, 115);  
 ellipse.setFill(Color.*YELLOW*);  
 gr.getChildren().add(ellipse);  
  
 Ellipse ellipse1 = new Ellipse(385,130,12, 25);  
 ellipse1.setFill(Color.*BLACK*);  
 ellipse1.getTransforms().add(new Rotate(-20, 385,130));  
 gr.getChildren().add(ellipse1);  
  
 Ellipse ellipse2 = new Ellipse(388,123,5, 8);  
 ellipse2.setFill(Color.*WHITE*);  
 ellipse2.getTransforms().add(new Rotate(-20, 385,130));  
 gr.getChildren().add(ellipse2);  
  
 Rectangle rect = new Rectangle(  
 80,270,  
 450,230  
 );  
 rect.setArcWidth(300);  
 rect.setArcHeight(200);  
 rect.setFill(Color.*YELLOW*);  
 gr.getChildren().add(rect);  
  
 Polygon polygon = new Polygon();  
 polygon.getPoints().addAll(new Double[]{  
 100.0, 330.0,  
 122.0, 222.0,  
 145.0, 208.0,  
 205.0, 280.0});  
 polygon.setFill(Color.*YELLOW*);  
 polygon.setStroke(Color.*YELLOW*);  
 polygon.setStrokeWidth(10);  
 polygon.setStrokeLineCap(StrokeLineCap.*ROUND*);  
 polygon.setStrokeLineJoin(StrokeLineJoin.*ROUND*);  
 gr.getChildren().add(polygon);  
  
 Polygon polygon1 = new Polygon();  
 polygon1.getPoints().addAll(new Double[]{  
 390.0, 218.0,  
 433.0, 182.0,  
 472.0, 144.0,  
 493.0, 152.0,  
 533.0, 148.0,  
 506.0, 196.0,  
 498.0, 237.0,  
 494.0, 240.0});  
 polygon1.setFill(Color.*ORANGE*);  
 polygon1.setStroke(Color.*ORANGE*);  
 polygon1.setStrokeWidth(10);  
 polygon1.setStrokeLineCap(StrokeLineCap.*ROUND*);  
 polygon1.setStrokeLineJoin(StrokeLineJoin.*ROUND*);  
 gr.getChildren().add(polygon1);  
  
 Polygon polygon2 = new Polygon();  
 polygon2.getPoints().addAll(new Double[]{  
 439.0, 218.0,  
 505.0, 192.0,  
 503.0, 219.0});  
 polygon2.setFill(Color.*RED*);  
 polygon2.setStroke(Color.*RED*);  
 polygon2.setStrokeWidth(10);  
 polygon2.setStrokeLineCap(StrokeLineCap.*ROUND*);  
 polygon2.setStrokeLineJoin(StrokeLineJoin.*ROUND*);  
 gr.getChildren().add(polygon2);  
  
  
 TranslateTransition translateTransition = new TranslateTransition(Duration.*millis*(2000), gr);  
 translateTransition.setFromX(50);  
 translateTransition.setToX(350);  
 translateTransition.setCycleCount(2);  
 translateTransition.setAutoReverse(true);  
 translateTransition.play();  
  
 ScaleTransition scaleTransition = new ScaleTransition(Duration.*millis*(2000), gr);  
 scaleTransition.setToX(320);  
 scaleTransition.setToY(320);  
 scaleTransition.setFromX(0.2);  
 scaleTransition.setFromY(0.2);  
 scaleTransition.setToX(0.5);  
 scaleTransition.setToY(0.5);  
 scaleTransition.setCycleCount(2);  
 scaleTransition.setAutoReverse(true);  
 scaleTransition.play();  
  
 RotateTransition rotForArc1 =  
 new RotateTransition(Duration.*millis*(500), gr);  
 rotForArc1.setByAngle(20f);  
 rotForArc1.setCycleCount(20);  
 rotForArc1.setAutoReverse(true);  
 rotForArc1.play();  
  
 primaryStage.show();  
 }  
  
  
 public static void main(String[] args) {  
 *launch*(args);  
 }  
}

**Результат**

****