

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-02  Щербатюк Ярослав Сергійович  (*прізвище, ім’я, по батькові*)  варіант № 14 |  |  | Зарахована  “\_\_\_\_” “\_\_\_\_\_\_\_\_\_\_\_\_” 20\_\_\_ р.  викладачем  Шкурат Оксаною Сергіївною (*прізвище, ім’я, по батькові*) |

Київ 2023

**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:

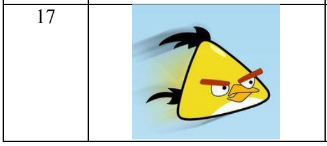
1) переміщення;

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**

|  |
| --- |
| Lab3.java |
| package lab3;  import javafx.animation.\*;  import javafx.application.Application;  import javafx.scene.\*;  import javafx.scene.paint.Color;  import javafx.scene.shape.\*;  import javafx.stage.\*;  import javafx.util.Duration;  public class lab3 extends Application {      public static void main(String args[]) {          launch(args);      }      @Override      public void start(Stage primaryStage) {          Group root = new Group();          Scene scene = new Scene(root, 1200, 600);          // Upper tail          MoveTo moveTo = new MoveTo(660, 160);          QuadCurveTo upper\_tail1 = new QuadCurveTo(650, 105, 560, 105);          QuadCurveTo upper\_tail2 = new QuadCurveTo(585, 110, 580, 125);          QuadCurveTo upper\_tail3 = new QuadCurveTo(440, 125, 390, 205);          QuadCurveTo upper\_tail4 = new QuadCurveTo(450, 165, 505, 165);          QuadCurveTo upper\_tail5 = new QuadCurveTo(480, 170, 450, 205);          QuadCurveTo upper\_tail6 = new QuadCurveTo(480, 190, 520, 180);          QuadCurveTo upper\_tail7 = new QuadCurveTo(525, 190, 530, 200);          Path upper\_tail = new Path();          upper\_tail.setStroke(Color.BLACK);          upper\_tail.setFill(Color.BLACK);          upper\_tail.getElements().addAll(moveTo, upper\_tail1, upper\_tail2, upper\_tail3, upper\_tail4, upper\_tail5,                  upper\_tail6, upper\_tail7);          root.getChildren().add(upper\_tail);          // Lower tail          MoveTo moveTo0 = new MoveTo(520, 360);          QuadCurveTo lower\_tail1 = new QuadCurveTo(470, 350, 450, 350);          QuadCurveTo lower\_tail2 = new QuadCurveTo(465, 360, 470, 365);          QuadCurveTo lower\_tail3 = new QuadCurveTo(380, 360, 340, 405);          QuadCurveTo lower\_tail4 = new QuadCurveTo(400, 375, 465, 385);          QuadCurveTo lower\_tail5 = new QuadCurveTo(455, 400, 440, 405);          QuadCurveTo lower\_tail6 = new QuadCurveTo(480, 410, 520, 395);          Path lower\_tail = new Path();          lower\_tail.setStroke(Color.BLACK);          lower\_tail.setFill(Color.BLACK);          lower\_tail.getElements().addAll(moveTo0, lower\_tail1, lower\_tail2, lower\_tail3, lower\_tail4, lower\_tail5,                  lower\_tail6);          root.getChildren().add(lower\_tail);          // Triangle part          MoveTo moveTo1 = new MoveTo(590, 155);          QuadCurveTo triangle\_line1 = new QuadCurveTo(510, 290, 470, 465);          QuadCurveTo triangle\_angle1 = new QuadCurveTo(470, 485, 500, 490);          QuadCurveTo triangle\_line2 = new QuadCurveTo(630, 500, 900, 420);          QuadCurveTo triangle\_angle2 = new QuadCurveTo(920, 405, 900, 380);          QuadCurveTo triangle\_line3 = new QuadCurveTo(770, 230, 660, 160);          QuadCurveTo triangle\_angle3 = new QuadCurveTo(610, 125, 590, 155);          Path triangle = new Path();          triangle.setStrokeWidth(7);          triangle.setStroke(Color.BLACK);          triangle.setFill(Color.YELLOW);          triangle.getElements().addAll(moveTo1, triangle\_line1, triangle\_angle1, triangle\_line2, triangle\_angle2,                  triangle\_line3, triangle\_angle3);          root.getChildren().add(triangle);          // Left eye          MoveTo moveTo2 = new MoveTo(630, 340);          QuadCurveTo left\_eye1 = new QuadCurveTo(625, 360, 630, 370);          QuadCurveTo left\_eye2 = new QuadCurveTo(650, 395, 700, 373);          QuadCurveTo left\_eye3 = new QuadCurveTo(715, 365, 710, 345);          Path left\_eye = new Path();          left\_eye.setStrokeWidth(4);          left\_eye.setStroke(Color.BLACK);          left\_eye.setFill(Color.WHITE);          left\_eye.getElements().addAll(moveTo2, left\_eye1, left\_eye2, left\_eye3);          root.getChildren().add(left\_eye);          // Left apple          Circle left\_apple = new Circle(690, 350, 7);          left\_apple.setFill(Color.BLACK);          root.getChildren().add(left\_apple);          // Left eyebrow          MoveTo moveTo3 = new MoveTo(575, 335);          QuadCurveTo left\_eyebrow1 = new QuadCurveTo(565, 325, 580, 315);          LineTo left\_eyebrow2 = new LineTo(715, 330);          QuadCurveTo left\_eyebrow3 = new QuadCurveTo(725, 338, 715, 345);          LineTo left\_eyebrow4 = new LineTo(575, 335);          Path left\_eyebrow = new Path();          left\_eyebrow.setStroke(Color.BROWN);          left\_eyebrow.setFill(Color.BROWN);          left\_eyebrow.getElements().addAll(moveTo3, left\_eyebrow1, left\_eyebrow2, left\_eyebrow3, left\_eyebrow4);          root.getChildren().add(left\_eyebrow);          // Right eye          MoveTo moveTo4 = new MoveTo(760, 330);          QuadCurveTo right\_eye1 = new QuadCurveTo(770, 365, 808, 360); //770 365 805 355          QuadCurveTo right\_eye2 = new QuadCurveTo(835, 340, 825, 305);          Path right\_eye = new Path();          right\_eye.setStrokeWidth(4);          right\_eye.setStroke(Color.BLACK);          right\_eye.setFill(Color.WHITE);          right\_eye.getElements().addAll(moveTo4, right\_eye1, right\_eye2);          root.getChildren().add(right\_eye);          // Right eyebrow          MoveTo moveTo5 = new MoveTo(750, 325);          LineTo right\_eyebrow1 = new LineTo(860, 270);          QuadCurveTo right\_eyebrow2 = new QuadCurveTo(875, 275, 870, 285);          LineTo right\_eyebrow3 = new LineTo(758, 340);          QuadCurveTo right\_eyebrow4 = new QuadCurveTo(740, 330, 750, 325);          Path right\_eyebrow = new Path();          right\_eyebrow.setStroke(Color.BROWN);          right\_eyebrow.setFill(Color.BROWN);          right\_eyebrow.getElements().addAll(moveTo5, right\_eyebrow1, right\_eyebrow2, right\_eyebrow3, right\_eyebrow4);          root.getChildren().add(right\_eyebrow);          // Right apple          Circle right\_apple = new Circle(810, 330, 7);          right\_apple.setFill(Color.BLACK);          root.getChildren().add(right\_apple);          // White bottom          MoveTo moveTo5\_5 = new MoveTo(630, 478);          QuadCurveTo white\_bottom\_up = new QuadCurveTo(750, 350, 875, 421);          QuadCurveTo white\_bottom\_down = new QuadCurveTo(750, 460, 630, 478);          Path white\_bottom = new Path();          white\_bottom.setStrokeWidth(4);          white\_bottom.setStroke(Color.WHITE);          white\_bottom.setFill(Color.WHITE);          white\_bottom.getElements().addAll(moveTo5\_5, white\_bottom\_up, white\_bottom\_down);          root.getChildren().add(white\_bottom);          // Beak upper          MoveTo moveTo6 = new MoveTo(740, 345);          CubicCurveTo beak1 = new CubicCurveTo(780, 345, 790, 370, 810, 360);          QuadCurveTo beak2 = new QuadCurveTo(820, 355, 835, 365);          QuadCurveTo beak3 = new QuadCurveTo(840, 375, 820, 375);          LineTo beak4 = new LineTo(760, 375);          QuadCurveTo beak5 = new QuadCurveTo(750, 375, 730, 390);          QuadCurveTo beak6 = new QuadCurveTo(715, 395, 735, 350);          QuadCurveTo beak7 = new QuadCurveTo(737, 345, 740, 345);          Path beak\_upper = new Path();          beak\_upper.setStrokeWidth(4);          beak\_upper.setStroke(Color.BLACK);          beak\_upper.setFill(Color.rgb(245, 192, 2));          beak\_upper.getElements().addAll(moveTo6, beak1, beak2, beak3, beak4, beak5, beak6, beak7);          root.getChildren().add(beak\_upper);          // Beak lower          MoveTo moveTo7 = new MoveTo(720, 397);          QuadCurveTo beak8 = new QuadCurveTo(755, 395, 760, 375);          LineTo beak9 = new LineTo(800, 375);          LineTo beak10 = new LineTo(760, 410);          QuadCurveTo beak11 = new QuadCurveTo(755, 413, 750, 410);          QuadCurveTo beak12 = new QuadCurveTo(710, 400, 720, 397);          Path beak\_lower = new Path();          beak\_lower.setStrokeWidth(4);          beak\_lower.setStroke(Color.BLACK);          beak\_lower.setFill(Color.rgb(245, 192, 2));          beak\_lower.getElements().addAll(moveTo7, beak8, beak9, beak10, beak11, beak12);          root.getChildren().add(beak\_lower);          // Reflection          MoveTo moveTo8 = new MoveTo(655, 178);          QuadCurveTo reflection1 = new QuadCurveTo(700, 240, 728, 230);          QuadCurveTo reflection2 = new QuadCurveTo(680, 180, 655, 178);          Path reflection = new Path();          reflection.setStroke(Color.rgb(250, 245, 164));          reflection.setFill(Color.rgb(250, 245, 164));          reflection.getElements().addAll(moveTo8, reflection1, reflection2);          root.getChildren().add(reflection);          // Animation          int cycleCount = 2;          int time = 2000;          RotateTransition rotateTransition = new RotateTransition(Duration.millis(time), root);          rotateTransition.setByAngle(-25f);          rotateTransition.setCycleCount(cycleCount);          rotateTransition.setAutoReverse(true);          TranslateTransition translateTransition = new TranslateTransition(Duration.millis(time), root);          translateTransition.setFromX(50);          translateTransition.setToX(1000);          translateTransition.setFromY(50);          translateTransition.setToY(-150);          translateTransition.setCycleCount(cycleCount);          translateTransition.setAutoReverse(true);          ScaleTransition scaleTransition = new ScaleTransition(Duration.millis(time), root);          scaleTransition.setToX(0.2);          scaleTransition.setToY(0.2);          scaleTransition.setCycleCount(cycleCount);          scaleTransition.setAutoReverse(true);          ParallelTransition parallelTransition = new ParallelTransition();          parallelTransition.getChildren().addAll(                  rotateTransition,                  scaleTransition,                  translateTransition          );          parallelTransition.setCycleCount(Timeline.INDEFINITE);          parallelTransition.play();          // End of animation          primaryStage.setResizable(false);          primaryStage.setTitle("Lab 3");          primaryStage.setScene(scene);          primaryStage.show();      }  } |

**Результат**

