import java.util.Arrays;

public class Statistics {

double sum(double[] a) {

double sum = 0;

for (double number : a) {

sum += number;

}

return sum;

}

double mean(double[] a) {

return (sum(a) / a.length);

}

int[] prio(int[] a) {

int max = 0;

int maxIndex = 0;

for (int i = 0; i < a.length; i++) { // store the maximum value and its index in a

if (a[i] > max) {

maxIndex = i;

max = a[i];

}

}

a[maxIndex] = a[0]; // replaces max element with first element

a[0] = max; // replaces first element with max element

return a;

}

}

public class Palindrome {

String reverse(String s) {

if (s.length() <= 1) {

return s;

}

return reverse(s.substring(1)) + s.charAt(0);

}

boolean isPalindrome(String s) {

return s.equals(reverse(s));

}

}

import java.util.ArrayList;

public class Humor {

}

class Joke {

String name;

String text;

int duration;

double funFactor;

Joke() {

name = "best your mom joke ever";

text = "what does your mother see when she steps on the weight scale?

her own phone number.";

duration = 6;

funFactor = 10.0;

}

Joke(String name, String text, int duration, double funFactor) {

this.name = name;

this.text = text;

this.duration = duration;

this.funFactor = funFactor;

}

}

class InsultingJoke extends Joke {

String socialGroupName;

InsultingJoke(String name, String text, int duration,

double funFactor, String socialGroupName) {

super(name, text, duration, funFactor);

this.socialGroupName = socialGroupName;

}

}

class Conference {

ArrayList<Joke> jokes = new ArrayList<>();

void add(Joke j) {

jokes.add(j);

}

int totalDuration() {

int totalDuration = 0;

for (Joke joke : jokes) {

totalDuration += joke.duration;

}

return totalDuration;

}

double calculateFunniness(Joke j) {

return (j.funFactor / j.duration);

}

double calculateFunniness(InsultingJoke j) {

return ((j.funFactor \* 1.5) / j.duration);

}

}

import java.util.\*;

import java.awt.\*;

import javax.swing.\*;

import javax.swing.Timer;

import java.awt.event.\*;

class Ballroom {

BallroomPanel ballroomPanel = new BallroomPanel();

Timer timer;

void createGUI() {

// create the GUI on the event thread.

SwingUtilities.invokeLater(new Runnable() {

@Override

public void run() {

final JFrame frame = new JFrame("Ballroom");

frame.add(ballroomPanel, BorderLayout.CENTER);

frame.setSize(600, 400);

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

frame.setVisible(true);

ballroomPanel.addBalls();

timer = new Timer(50, ballroomPanel);

timer.start();

}

});

}

public static void main(String[] a) {

new Ballroom().createGUI();

}

}

class BallroomPanel extends JPanel implements ActionListener {

Timer timer;

ArrayList<Ball> balls =

new ArrayList<Ball>(); // the list of monsters on the screen

void addBalls() {

// I moved the original ball to the middle, otherwise red and blinking aren't visible

balls.add(new Ball(300, 10));

balls.add(new Ball(380, 10));

balls.add(new RedBall(220, 10));

balls.add(new BlinkingBall(260, 10));

}

public void paintComponent(Graphics g) {

super.paintComponent(g);

for (Ball ball : balls) {

ball.draw(g);

}

}

public void actionPerformed(ActionEvent e) {

for (Ball ball : balls) {

ball.step();

}

repaint();

}

}

class Ball {

int radius;

int x; // x coordinate center (pixel coordinates)

int y; // y coordinate center (pixel coordinates)

int fallStep;

Color ballColor;

Ball() {

// location extremely top left

this(0, 0);

}

Ball(int x, int y) {

this.x = x;

this.y = y;

radius = 20;

fallStep = 20;

ballColor = Color.BLUE;

}

// update the ball because a time step has passed

void step() {

y += fallStep;

}

void draw(Graphics g) {

// draw ball

g.setColor(ballColor);

g.fillOval(x - radius / 2, y - radius / 2, radius, radius);

}

}

class RedBall extends Ball{

RedBall(int x, int y) {

super(x, y);

ballColor = Color.RED;

fallStep = fallStep \* 2;

}

}

class BlinkingBall extends Ball {

Timer timer;

BlinkingBall(int x, int y) {

super(x, y);

timer = new Timer(200, e -> {

if (ballColor.equals(Color.BLUE)){

ballColor = Color.RED;

} else {

ballColor = Color.BLUE;

}

});

timer.start();

}

}