import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

public class SnakeGame extends JFrame {

private static final int TILE\_SIZE = 30;

private static final int GRID\_WIDTH = 20;

private static final int GRID\_HEIGHT = 20;

private static final int DELAY = 150;

private Timer timer;

private int[] snakeX = new int[GRID\_WIDTH \* GRID\_HEIGHT];

private int[] snakeY = new int[GRID\_WIDTH \* GRID\_HEIGHT];

private int snakeLength;

private int foodX, foodY;

private char direction = 'R';

private boolean running = false;

public SnakeGame() {

initGame();

addKeyListener(new KeyAdapter() {

@Override

public void keyPressed(KeyEvent e) {

switch (e.getKeyCode()) {

case KeyEvent.VK\_UP:

if (direction != 'D') direction = 'U';

break;

case KeyEvent.VK\_DOWN:

if (direction != 'U') direction = 'D';

break;

case KeyEvent.VK\_LEFT:

if (direction != 'R') direction = 'L';

break;

case KeyEvent.VK\_RIGHT:

if (direction != 'L') direction = 'R';

break;

}

}

});

setTitle("Snake Game");

setSize(GRID\_WIDTH \* TILE\_SIZE, GRID\_HEIGHT \* TILE\_SIZE);

setResizable(true);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setLocationRelativeTo(null);

setVisible(true);

}

private void initGame() {

running = true;

snakeLength = 3;

for (int i = 0; i < snakeLength; i++) {

snakeX[i] = 5 - i;

snakeY[i] = 5;

}

spawnFood();

timer = new Timer(DELAY, e -> gameLoop());

timer.start();

}

private void spawnFood() {

foodX = (int) (Math.random() \* GRID\_WIDTH);

foodY = (int) (Math.random() \* GRID\_HEIGHT);

}

private void gameLoop() {

if (running) {

move();

checkFoodCollision();

checkSelfCollision();

repaint();

} else {

timer.stop();

}

}

private void move() {

for (int i = snakeLength; i > 0; i--) {

snakeX[i] = snakeX[i - 1];

snakeY[i] = snakeY[i - 1];

}

switch (direction) {

case 'U': snakeY[0]--; break;

case 'D': snakeY[0]++; break;

case 'L': snakeX[0]--; break;

case 'R': snakeX[0]++; break;

}

if (snakeX[0] < 0 || snakeY[0] < 0 || snakeX[0] >= GRID\_WIDTH || snakeY[0] >= GRID\_HEIGHT) {

running = false;

}

}

private void checkFoodCollision() {

if (snakeX[0] == foodX && snakeY[0] == foodY) {

snakeLength++;

spawnFood();

}

}

private void checkSelfCollision() {

for (int i = 1; i < snakeLength; i++) {

if (snakeX[0] == snakeX[i] && snakeY[0] == snakeY[i]) {

running = false;

}

}

}

@Override

public void paint(Graphics g) {

super.paint(g);

if (running) {

// Draw food

g.setColor(Color.RED);

g.fillRect(foodX \* TILE\_SIZE, foodY \* TILE\_SIZE, TILE\_SIZE, TILE\_SIZE);

// Draw snake

for (int i = 0; i < snakeLength; i++) {

g.setColor(i == 0 ? Color.GREEN : Color.BLUE);

g.fillRect(snakeX[i] \* TILE\_SIZE, snakeY[i] \* TILE\_SIZE, TILE\_SIZE, TILE\_SIZE);

}

} else {

// Game Over

g.setColor(Color.BLACK);

g.setFont(new Font("Arial", Font.BOLD, 50));

g.drawString("Game Over", getWidth() / 4, getHeight() / 2);

}

}

public static void main(String[] args) {

SwingUtilities.invokeLater(SnakeGame::new);

}

}