import java.awt.\*;

import java.awt.event.\*;

import java.awt.image.\*;

import javax.swing.\*;

import java.io.\*;

import javax.imageio.\*;

class Game extends JFrame {

    Image img[] = new Image[20];

    final int p1up = 0;

    final int p1right = 1;

    final int p1left = 2;

    final int p1down = 3;

    final int bg1 = 4;

    final int bg2 = 5;

    final int bg3 = 6;

    final int bg4 = 7;

    final int bg5 = 8;

    final int bg6 = 9;

    final int meteor1 = 10;

    final int meteor2 = 11;

    final int meteor3 = 12;

    final int meteor4 = 13;

    final int p2up = 14;

    final int p2right = 15;

    final int p2left = 16;

    final int p2down = 17;

    int p1x = 0, p1y = -50, p1Sprite = p1up;

    int p2x = 1000, p2y = -50, p2Sprite = p2up;

    int moveSpeed = 20, meteorSpeed[] = new int[4];

    int meteorTrajectory[] = new int[4];

    Timer t;

    int score = 0;

    // funcao para gerar numeros aleatorios

    public int getRandomNumber(int min, int max) {

        return (int) ((Math.random() \* (max - min)) + min);

    }

    int rand[] = new int[4];

    // Seleciona um background aleatorio

    int bgSelected = getRandomNumber(bg1, bg6);

    GameMap map = new GameMap();

    // função para gerar um retangulo

    public Rectangle getBounds(int x, int y, int width, int height) {

        return new Rectangle(x, y, width, height);

    }

    int mWidth[] = new int[4];

    int mHeight[] = new int[4];

    int pWidth[] = new int[2];

    int pHeight[] = new int[2];

    int py[] = new int[2];

    final int width = 1800;

    final int height = 900;

    // Classe que adicionara as imagens no jogo

    class GameMap extends JPanel {

        GameMap() {

            // zerando trajetorias e geranod velocidade e localização aleatoria para os

            // meteoros

            for (int i = 0; i < 4; i++) {

                meteorTrajectory[i] = 0;

                meteorSpeed[i] = getRandomNumber(10, 20);

                rand[i] = getRandomNumber(0, width);

            }

            setPreferredSize(new Dimension(width, height));

            // leitura das imagens

            try {

                img[p1up] = ImageIO.read(new File("img/player1/p1up.png"));

                img[p1right] = ImageIO.read(new File("img/player1/p1right.png"));

                img[p1down] = ImageIO.read(new File("img/player1/p1down.png"));

                img[p1left] = ImageIO.read(new File("img/player1/p1left.png"));

                img[bg1] = ImageIO.read(new File("img/bg/bg1.jpg"));

                img[bg2] = ImageIO.read(new File("img/bg/bg2.jpg"));

                img[bg3] = ImageIO.read(new File("img/bg/bg3.jpg"));

                img[bg4] = ImageIO.read(new File("img/bg/bg4.jpg"));

                img[bg5] = ImageIO.read(new File("img/bg/bg5.jpg"));

                img[bg6] = ImageIO.read(new File("img/bg/bg6.jpg"));

                img[meteor1] = ImageIO.read(new File("img/meteor/meteor1.png"));

                img[meteor2] = ImageIO.read(new File("img/meteor/meteor2.png"));

                img[meteor3] = ImageIO.read(new File("img/meteor/meteor3.png"));

                img[meteor4] = ImageIO.read(new File("img/meteor/meteor4.png"));

                img[p2up] = ImageIO.read(new File("img/player2/p2up.png"));

                img[p2right] = ImageIO.read(new File("img/player2/p2right.png"));

                img[p2down] = ImageIO.read(new File("img/player2/p2down.png"));

                img[p2left] = ImageIO.read(new File("img/player2/p2left.png"));

            } catch (IOException e) {

                JOptionPane.showMessageDialog(this, "A imagem não pode ser carregada!\n" + e, "Erro",

                        JOptionPane.ERROR\_MESSAGE);

                System.exit(1);

            }

        }

        public void paintComponent(Graphics g) {

            super.paintComponent(g);

            // background

            g.drawImage(img[bgSelected], 0, 0, getSize().width, getSize().height, this);

            // Players

            g.drawImage(img[p1Sprite], p1x, getSize().height - img[p1Sprite].getHeight(this) + p1y, this);

            g.drawImage(img[p2Sprite], p2x, getSize().height - img[p2Sprite].getHeight(this) + p2y, this);

            // Meteoros

            g.drawImage(img[meteor1], rand[0], meteorTrajectory[0], this);

            g.drawImage(img[meteor2], rand[1], meteorTrajectory[1], this);

            g.drawImage(img[meteor3], rand[2], meteorTrajectory[2], this);

            g.drawImage(img[meteor4], rand[3], meteorTrajectory[3], this);

            // altura dos meteoros

            mHeight[0] = img[meteor1].getHeight(this);

            mHeight[1] = img[meteor2].getHeight(this);

            mHeight[2] = img[meteor3].getHeight(this);

            mHeight[3] = img[meteor4].getHeight(this);

            // largura dos meteoros

            mWidth[0] = img[meteor1].getWidth(this);

            mWidth[1] = img[meteor2].getWidth(this);

            mWidth[2] = img[meteor3].getWidth(this);

            mWidth[3] = img[meteor4].getWidth(this);

            // altura dos jogadores

            pHeight[0] = img[p1Sprite].getHeight(this);

            pHeight[1] = img[p2Sprite].getHeight(this);

            // largura dos jogadores

            pWidth[0] = img[p1Sprite].getWidth(this);

            pWidth[1] = img[p2Sprite].getWidth(this);

            // posY dos jogadores

            py[0] = getSize().height - img[p1Sprite].getHeight(this) + p1y;

            py[1] = getSize().height - img[p2Sprite].getHeight(this) + p2y;

            Toolkit.getDefaultToolkit().sync();

        }

    }

    Game() {

        super("Game");

        setDefaultCloseOperation(EXIT\_ON\_CLOSE);

        add(map);

        pack();

        setVisible(true);

        t = new Timer(1000, new ActionListener() {

            public void actionPerformed(ActionEvent ae) {

                score++;

                for (int i = 0; i < 4; i++) {

                    // faz os meteoros voltarem a cair com velocidade aumentanda e de posicoes

                    // aleatorias

                    if (meteorTrajectory[i] >= height) {

                        meteorTrajectory[i] = 0;

                        meteorSpeed[i] += getRandomNumber(0, 5);

                        moveSpeed += getRandomNumber(0, 1);

                        for (int j = 0; j < 4; j++) {

                            rand[i] = getRandomNumber(0, width);

                        }

                    } else {

                        meteorTrajectory[i] += meteorSpeed[i]; // senao chegou no limite da tela ele continua caindo

                        // testa se os meteoros bateram nos jogadores, se baterem o jogo acaba e mostra

                        // a pontuação

                        if (getBounds(rand[i], meteorTrajectory[i], mWidth[i], mHeight[i])

                                .intersects(getBounds(p1x, py[0], pWidth[0], pHeight[0]))) {

                            JOptionPane.showMessageDialog(null, "MARCOU " + score + " PONTOS ANTES DE MORRER");

                            System.exit(1);

                        } else if (getBounds(rand[i], meteorTrajectory[i], mWidth[i], mHeight[i])

                                .intersects(getBounds(p2x, py[1], pWidth[1], pHeight[1]))) {

                            JOptionPane.showMessageDialog(null, "MARCOU " + score + " PONTOS ANTES DE MORRER");

                            System.exit(1);

                        }

                    }

                    repaint();

                }

                repaint();

            }

        });

        t.start();

        // Tratar teclado

        addKeyListener(new KeyAdapter() {

            public void keyPressed(KeyEvent key) {

                // Esc para sair

                if (key.getKeyCode() == KeyEvent.VK\_ESCAPE) {

                    System.exit(1);

                }

                // WASD para movimento do jogador

                if (key.getKeyCode() == KeyEvent.VK\_W) {

                    p1Sprite = p1up;

                    p1y -= moveSpeed;

                } else if (key.getKeyCode() == KeyEvent.VK\_S) {

                    p1Sprite = p1down;

                    p1y += moveSpeed;

                } else if (key.getKeyCode() == KeyEvent.VK\_D) {

                    p1Sprite = p1right;

                    p1x += moveSpeed;

                } else if (key.getKeyCode() == KeyEvent.VK\_A) {

                    p1Sprite = p1left;

                    p1x -= moveSpeed;

                }

                repaint();

            }

        });

    }

    static public void main(String[] args) {

        Game game = new Game();

    }

}