Shape, square

Description automatically generated

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**UNIVERSITY OF TRANSPOST AND COMMUNICATIONS**

Logo

Description automatically generated

**BÁO CÁO BÀI TẬP LỚN**

**MÔN CÔNG NGHỆ JAVA**

***ĐỀ TÀI: Lập trình game rắn săn mồi***

**Giảng viên hướng dẫn: Đào Thị Lệ Thủy**

**Sinh viên thực hiện: Trần Trung Hiếu – Lê Hồng Phong**

**Lớp: CNTT4**

**Năm 2022**

**I : Tổng quan về đề tài**

**1 : Giới thiệu về đề tài**

-Qua nghiên cứu và tìm hiểu ta nhận thấy game Rắn săn mồi là game cổ điện xuất hiện vào năm 1997 trên Nokia 6610 là những ô vuông xếp liên tiếp nhau di chuyển trên một màn hình màu xanh đơn giản, nhưng Rắn săn mồi đã xây dựng rất thành công tên tuổi của mình. Với 400 triệu bản được xuất xưởng và đến hiện tại đã là phiên bản thứ 8. Game có sự hấp đẫn đối với người chơi. Do đó nhóm em quyết định xây dựng game dựa trên những ý tưởng từ xưa cũ nhưng chưa bao giờ là mất đi chất của tựa game dã tạo nên tuổi thơ của một thế hệ.

**2 : Chức năng của chương trình**

-Rắn có thể đi lên xuống trái phải theo góc 90o mỗi khi chuyển hướng

-Có giao diện khi mới bắt đầu chạy game, phải bấm “Start” thì game mới chạy để chơi

-Khi đâm vào thân hay tường có thể bấm “Enter” để có thể chơi lại

-Có âm thanh khi chơi game

**3 : Yêu cầu chương trình**

**-**Chương trình được thiết kế gồm 3 file

+File GameScreen là file dùng để chạy game

+File FrameScreen là file dùng để thiết kế phần giao diện trước khi vào game và hiển thị màn hình chơi Game

+File GamePanel dùng để kiểm soát các chức năng của rắn (di chuyển, hình ảnh rắn chạy, kiểm tra rắn có đâm không, hiện Game Over khi đâm vào người hoặc vào tường…)

-Giao diện của cửa sổ trước khi vào Game bao gồm hình rắn và một nút start để bắt đầu Game :

+Kích thước 500\*500px

+Nền hình rắn

+Button Start có kích thước 100\*50px và đặt ở tọa độ x=200,y=300

Shape

Description automatically generated

-Cửa sổ chính chơi game :

+Có kích thước 1280\*720 px

+Được bố trí như ma trận mỗi ô kích thước 30

+Chữ Start được đặt ở góc trên cửa sổ có size 75px và có tọa độ x=200

,y=300

Text

Description automatically generated with low confidence

+Sau khi đâm vào tường hay thân sẽ hiện ra màn hình Game Over và Press Enter to restart

+Chữ Game Over có màu = RED font size 75px và có tọa độ x =(1280 – stringWidth( “GameOver”))/2, y = 720/2 để có thể căn ở ngay chính giữa màn hình

+Chữ Press Enter to restart game có màu = GREEN, font size 75px và có tọa độ là x = (1280 – stringWidth(“Press Enter to restart game”))/2-120, y = 720-200.



+Các chữ Score, Game Over, Press Enter to restart game có font Ink Free và font size 75px.

**II : Phân tích thiết kế chương trình**

**Các đối tượng và chức năng**

\*Class GamePannel () : Chứa hàm main để chạy chương trình

\*Class GameFrame () : Dùng để thiết kế giao diện trước khi vào Game

\*Class GamePannel () : Chứa các phương thức để khởi tạo cũng như điều khiển chương trình :

**\*Biến :**

**-static final int SCREEN\_WIDTH :** Biến không đổi chiều dài của chương trình

**-static final int SCREEN\_HEIGHT :** Biến không đổi chiều rộng của chương trình

**-static final int UNIT\_SIZE = 30 :** kích thước 1 ô trong ma trận ta tạo là 30

**-static final int GAME\_UNIT :** Tổng unit của game

**-final int x[] = new int [GAME\_UNIT] :** mảng x có kích thước bằng game unit

**-final int y[] = new int [GAME\_UNIT] :** mảng y có kích thước bằng game unit

**-int appleEaten :** Biến đếm số điểm rắn ăn được

**-int appleX,appleY:** chỉ tọa độ của táo

**-boolean running = false :** biến để kiểm soát chương trình chạy và dừng

**\*Các phương thức :**

**-GamePanel() :** Phương thức khởi tạo khi ta gọi đến class GamePanel

+Tạo ra một cửa sổ bằng với SCREEN\_WIDTH và SCREEN\_HEIGHT

+Tiếp theo nó sẽ tạo nền cho chương trình là màu = BLACK

+AddKeyListener(newMyKeyAdapter) để có thể lấy các nút bấm từ người dùng từ đó có thể điều khiển rắn

+Cuối cùng chạy đến phương thức StartGame() để chạy chương trình

**-public void StartGame() :** Dùng để chạy chương trình

+playMusic(GTASAN.wav) : để chạy nhạc khi chương trình khởi động

+Gọi tới phương thức newApple() để có thể tạo ra táo cho chương trình

+Đặt running = true để chương trình chạy

+timer = new Timer(DELAY,this) dùng để chạy chương trình với DELAY là 75 để chương trình chạy không quá nhanh và mắt thường có thể theo được

+timer.start() để bắt đầu chạy chương trình

**-public void paintComponent (Graphics g) :** Dùng để reset khung hình khi rắn di chuyển

+super.paintComponent(g) : khi rắn di chuyển chương trình sẽ chạy tới phương thức paintComponent để có thể xóa đi khung hình trước đó của rắn để thay vào đó một khung hình mới khi rắn di chuyển

+Tiếp theo sẽ gọi tới phương thức draw(g) để có thể vẽ lại một vị trí mới của rắn

**-public void draw (Graphics g) :** Dùng để vẽ rắn và ghi điểm của người chơi

+Ta sẽ sử dụng cấu trúc if else để có thể kiểm tra xem chương trình có chạy hay không

+Nếu running = true thì ta sẽ chạy một vòng for từ i=0 cho đến i = bodypart là chiều dài của rắn, khi đó tương ứng với mỗi ô ta sẽ setColor cho rắn là ngẫu nhiên.

+Tiếp theo ta sẽ vẽ chữ “Score” có màu đỏ, font size 75px và font chữ là Ink Free để có thể hiển thị điểm của người chơi

+Nếu running = false thì ta sẽ gọi đến phương thức gameover(g)

**-public void newApple() :** Dùng để hiển thị táo mới khi rắn ăn

+Ta sẽ sử dụng phương thức nextInt để sinh số ngẫu nhiên cho tọa độ của táo từ đó ta có thể sinh táo ngẫu nhiên ở một trong các ô của ma trận

**-public void move () :** Dùng để kiểm soát đường di chuyển của rắn

+Ta sẽ sử dụng một vòng for có i=bodypart chạy tới i=0 để có thể gán x[i] = x[i-1] và y[i] = y[i-1] nghĩa là khi rắn di chuyển ta sẽ dịch mỗi ô trong thân của rắn dịch sang 1 đơn vị theo chiều của rắn đang chạy

+Tiếp theo ta sử dụng cấu trúc switch case để kiểm soát direction là hướng rắn đang chạy khi rẽ:

+case’U’ : Tức rắn vừa rẽ lên thì ta lấy tọa độ y của rắn trừ đi 1 ô là 1 UNIT\_SIZE

+case’L’ : Tức rắn vừa rẽ sang trái thì ta sẽ lấy tọa độ x của rắn trừ đi 1 UNIT SIZE tức 1 ô trong ma trận

+case’D’ : Tức rắn vừa rẽ xuống thì ta sẽ lấy tọa độ y của rắn cộng thêm 1 UNIT SIZE tức là 1 ô trong ma trận

+case’R’ : Tức rắn vừa rẽ sang phải thì ta sẽ lấy tọa độ x của rắn cộng thêm 1 UNIT SIZE tức là 1 ô của ma trận

**-public void checkApple() :** Dùng để kiểm soát khi rắn ăn táo

+Khi rắn ăn táo tức tọa độ của đầu rắn là x[0] và y[0] bằng với tọa độ của táo là appleX và appleY thì ta sẽ tăng appleseaten lên 1 đơn vị và bodypart lên 1 ô trong ma trận

+Cuối cùng sẽ gọi tới phương thức newApple() để có thể tạo ra táo mới

**-public void checkCollisons() :** Dùng để kiểm soát khi rắn đâm vào tường hoặc thân

+Trước hết ta sẽ dùng một vòng for để chạy từ i=bodypart đến i=0 để có thể kiểm soát toàn bộ thân rắn, khi x[0] = x[i] và y[0] = y[i] tức đầu rắn đâm vào người ta sẽ đặt running = false khi đó dòng chữ Game Over sẽ hiện ra

+Tiếp theo ta sẽ sử dụng cấu trúc if else để có thể kiểm soát rắn đâm vào tường

+Nếu x[0]<0 tức rắn đâm vào tường bên trái hoặc x[0] > SCREEN\_WIDTH tức rắn đâm vào tường bên phải hoặc y[0]<0 tức rắn đâm vào tường phía trên hoặc y[0] > SCREEN\_HEIGHT tức rắn đâm vào tường ở dưới thì ra sẽ đặt cho running = false để dừng chương trình và hiện lên dòng chữ Game Over

**-public void gameover(Graphics g) :** Dùng để vẽ chữ “Game Over” cũng như “Press Enter to restart”

+Dùng setColor để có thể lựa chọn màu cho chữ

+Dùng setFont để lựa chọn font chữ cũng như kích thước chữ

+Dùng drawstring để có thể đưa chuỗi ra màn hình cũng như vị trí ta cần đưa ra

**-public void actionPerformed(ActionEvent e) :** Dùng để gọi tới phương thức repaint khi rắn chạy

+Nếu running = true thì sẽ gọi tới phương thức move(), checkCollisons() cũng như checkApple() để rắn chạy

+Khi đó ta sẽ gọi tới phương thức repaint() chính là để chạy lại phương thức paintComponent(g) để có thể vẽ lại rắn ở một vị trí mới

**-public void keyPressed(KeyEvent e) :** Dùng để kiểm soát hướng đi của rắn

+Ta sử dụng switch case để có thể kiểm soát các đường đi của rắn

+case’VK\_LEFT’ tức là ta đang bấm mũi tên bên trái thì chương trình sẽ kiểm soát cho rắn sẽ không thể quay đầu đi hướng ngược lại là bên phải (Tương tự với 3 hướng còn lại là VK\_RIGHT,VK\_UP,VK\_DOWN)

+Ngoài ra ta cũng add event cho nút Enter(VK\_ENTER) : ta sẽ đặt lại tất cả từ điểm, tọa độ của thân rắn, bodypart để có thể chơi lại từ đầu(Chức năng restart)

**-public void playMusic(String musiclocation) :** Dùng để phát nhạc khi chương trình chạy

+audioInputStream dùng để lấy file âm thanh

+Sau đó ta sẽ sử dụng open và start để chạy file âm thanh

+loop dùng để chạy lại file nhạc khi chạy hết

**Sơ đồ tổ chức chương trình**

Chart

Description automatically generated

**III: Cài đặt chương trình**

**1.Gói Model :**

public class GameScreen {

public static void main(String[] args) {

new GameFrame();

}

}

**2.Gói view :**

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

public class GameFrame extends JFrame implements ActionListener{

JButton btnStart;

JLabel L1;

JButton resetButton;

GameFrame () {

this.setContentPane(new JLabel(new ImageIcon("sng.jpg")));

this.setLayout(new FlowLayout());

this.L1 = new JLabel();

this.add(L1);

this.setSize(500,500);

this.btnStart = new JButton();

this.btnStart.setBounds(200,300,100,50);

this.setTitle("Snake");

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

this.setLayout(null);

this.setSize(500,500);

this.setVisible(true);

this.add(btnStart);

this.setLocationRelativeTo(null);

this.btnStart.setText("Start");

this.btnStart.addActionListener(this);

this.setResizable(false);

}

@Override

public void actionPerformed(ActionEvent e) {

if (e.getSource()==btnStart) {

this.setVisible(false);

JFrame jFrame = new JFrame();

jFrame.add(new GamePanel());

jFrame.setTitle("Snake");

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

jFrame.setResizable(false);

jFrame.pack();

jFrame.setVisible(true);

jFrame.setLocationRelativeTo(null);

}

}

}

**3.Gói Controller :**

import java.awt.\*;

import java.awt.event.\*;

import javax.sound.sampled.AudioInputStream;

import javax.sound.sampled.AudioSystem;

import javax.sound.sampled.Clip;

import javax.swing.\*;

import java.io.File;

import java.util.Random;

public class GamePanel extends JPanel implements ActionListener{

static final int SCREEN\_WIDTH = 1280;

static final int SCREEN\_HEIGHT = 720;

static final int UNIT\_SIZE = 30;

static final int GAME\_UNIT = (SCREEN\_WIDTH\*SCREEN\_HEIGHT)/UNIT\_SIZE;

static final int DELAY = 75;

final int x[] = new int[GAME\_UNIT];

final int y[] = new int[GAME\_UNIT];

int bodyParts = 6;

int applesEaten;

int appleX;

int appleY;

char direction = 'R';

boolean running = false;

Timer timer;

Random random;

GamePanel() {

random = new Random();

this.setPreferredSize(new Dimension(SCREEN\_WIDTH,SCREEN\_HEIGHT));

this.setBackground(Color.black);

this.setFocusable(true);

this.addKeyListener(new MyKeyAdapter());

StartGame();

}

public void StartGame () {

playMusic("GTASAN.wav");

newApple();

running=true;

timer=new Timer(DELAY,this);

timer.start();

}

public void paintComponent (Graphics g) {

super.paintComponent(g);

draw(g);

}

public void draw (Graphics g) {

if (running) {

// for (int i = 0; i < SCREEN\_HEIGHT / UNIT\_SIZE; i++) {

// g.drawLine(i \* UNIT\_SIZE, 0, i \* UNIT\_SIZE, SCREEN\_HEIGHT);

// g.drawLine(0, i \* UNIT\_SIZE, SCREEN\_WIDTH, i \* UNIT\_SIZE);

// }

g.setColor(Color.red);

g.fillOval(appleX, appleY, UNIT\_SIZE, UNIT\_SIZE);

for (int i = 0; i < bodyParts; i++)

if (i == 0) {

g.setColor(Color.green);

g.fillRect(x[i], y[i], UNIT\_SIZE, UNIT\_SIZE);

}

else {

g.setColor(new Color(45, 180, 0));

g.setColor(new Color(random.nextInt(255),random.nextInt(255),random.nextInt(255)));

g.fillRect(x[i], y[i], UNIT\_SIZE, UNIT\_SIZE);

}

g.setColor(Color.red);

g.setFont(new Font("Ink Free",Font.BOLD,40));

FontMetrics metrics = getFontMetrics(g.getFont());

g.drawString("Score : "+applesEaten,(SCREEN\_WIDTH - metrics.stringWidth("Score : "+applesEaten))/2,g.getFont().getSize());

}

else {

gameover(g);

}

}

public void newApple() {

appleX= random.nextInt((int)SCREEN\_WIDTH/UNIT\_SIZE)\*UNIT\_SIZE;

appleY= random.nextInt((int)SCREEN\_HEIGHT/ UNIT\_SIZE)\*UNIT\_SIZE;

}

public void move (){

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

x[i] = x[i-1];

y[i] = y[i-1];

}

switch (direction) {

case 'U' :

y[0] = y[0] - UNIT\_SIZE;

break;

case 'D' :

y[0] = y[0] + UNIT\_SIZE;

break;

case 'L' :

x[0] = x[0] - UNIT\_SIZE;

break;

case 'R' :

x[0] = x[0] + UNIT\_SIZE;

break;

}

}

public void checkApple(){

if ((x[0] == appleX)&& (y[0] == appleY)) {

bodyParts++;

applesEaten++;

newApple();

}

}

public void checkCollisons(){

// If head collies with body

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

if ((x[0] == x[i]) && (y[0] == y[i])) {

running = false;

}

}

// If head touches left border

if (x[0] < 0) {

running = false;

}

// If head touches right border

if (x[0] > SCREEN\_WIDTH) {

running = false;

}

// If head touches top border

if (y[0] < 0) {

running = false;

}

// If head touches bottom border

if (y[0] > SCREEN\_HEIGHT) {

running = false;

}

if (!running) {

timer.stop();

}

}

public void gameover(Graphics g){

//Score

g.setColor(Color.red);

g.setFont( new Font("Ink Free",Font.BOLD,75));

FontMetrics metrics1= getFontMetrics(g.getFont());

g.drawString("Score: "+applesEaten,(SCREEN\_WIDTH -metrics1.stringWidth("Score: "+applesEaten))/2,g.getFont().getSize());

//Game Over

g.setColor(Color.red);

g.setFont( new Font("Ink Free",Font.BOLD,75));

FontMetrics metrics2= getFontMetrics(g.getFont());

g.drawString("Game Over",(SCREEN\_WIDTH -metrics2.stringWidth("Game Over"))/2,SCREEN\_HEIGHT/2);

g.setColor(Color.green);

g.drawString("Press Enter to restart game",(SCREEN\_WIDTH -metrics2.stringWidth("Press Enter to restart game")-120),SCREEN\_HEIGHT-200);

}

@Override

public void actionPerformed(ActionEvent e){

if(running){

move();

checkApple();

checkCollisons();

}

repaint();

}

public class MyKeyAdapter extends KeyAdapter{

@Override

public void keyPressed(KeyEvent e){

switch(e.getKeyCode()){

case KeyEvent.VK\_LEFT:

if(direction != 'R') {

direction='L';

}

break;

case KeyEvent.VK\_RIGHT:

if(direction != 'L') {

direction='R';

}

break;

case KeyEvent.VK\_UP:

if(direction != 'D') {

direction='U';

}

break;

case KeyEvent.VK\_DOWN:

if(direction != 'U') {

direction='D';

}

break;

case KeyEvent.VK\_ENTER:

if(!running){

running= true;

applesEaten=0;

bodyParts=6;

for (int i = 0; i < bodyParts; i++)

{

x[i]=0;

y[i]=0;

}

direction ='R';

timer.start();

move();

checkApple();

checkCollisons();

repaint();

}

break;

}

}

}

public void playMusic(String musiclocation){

try{

File musicPath = new File(musiclocation);

if(musicPath.exists()){

AudioInputStream audioInput= AudioSystem.getAudioInputStream(musicPath);

Clip clip=AudioSystem.getClip();

clip.open(audioInput);

clip.start();

clip.loop(Clip.LOOP\_CONTINUOUSLY);

}

else{

System.out.println("Cannot find the Audio File");

}

}catch (Exception ex){

ex.printStackTrace();

}

}

}

**IV: Giới thiệu giao diện thực hiện trò chơi**

**\*Giao diện chính của trò chơi bao gồm 2 cửa sổ**

-Cửa sổ đầu tiên khi ta chạy chương trình sẽ hiện ra để ta bấm “Start” trò chơi sẽ bắt đầu chạy

-Cửa sổ thứ 2 sẽ hiện ra khi ta bấm vào “Start”, cửa sổ để ta chơi sẽ hiện ra và cửa sổ mới đầu sẽ được ẩn đi

**\*Giao diện chơi game :**

**-Cấu tạo rắn :**

+Con rắn có độ dài là 6 ô trong ma trận và mỗi ô sẽ có kích thước là 30

+Rắn xuất phát từ tọa độ x=0,y=0 và có màu ngẫu nhiên khi chạy

**-Táo :**

+Táo được tạo ngẫu nhiên 1 trong các ô ở ma trận

+Kích thước của táo là 1 ô trong ma trận có kích thước 30

+Táo có màu đỏ và sẽ được hiển thị ngẫu nhiên ở một nơi khác khi rắn ăn

**-Màn hình Game Over**

+Màn hình Game Over sẽ được hiện ra khi rắn đâm vào tường hoặc đâm vào thân

+Màn hình Game Over sẽ bao gồm “Score” là điểm của người chơi khi thua, chữ “Game Over” có màu đỏ và dòng chữ “Press Enter to restart” để người chơi có thể bắt đầu lại nếu muốn

**V: Kết luận và đánh giá**

**1.Kết quả đạt được**

-Tạo được Snake game có thể chơi

-Tạo được khá đầy đủ các chức năng cơ bản của 1 game rắn :

+Có giao diện khi mới vào game

+Có các chức năng di chuyển ăn táo cơ bản

+Có nút bấm để chơi lại khi thua

**2.Hạn chế**

-Chưa tạo được chức năng pause game khi đang chơi

-Chưa đặt giao diện mới vào game và chơi game trên cùng 1 cửa sổ

-Chưa điểu chỉnh độ phân giải của game ở giao diện mà phải chỉnh ở trong code

**Tài liệu tham khảo :** Youtube và Stackoverflow