MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE , GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Skills Based Mini Project Report

on

Snake Game

Submitted By:

Vinayak Choubey

0901CA211068

Mentor:

Dr. Anshu Chaturvedi

(Professor)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE

GWALIOR - 474005 (MP) est. 1957

JANUARY - JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE , GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

CERTIFICATE

This is certified that **Vinayak Choubey (0901CA211068)** has submitted the project report titled **Snake game** under the mentorship of **Dr. Anshu Chaturvedi (Professor)**, as the skill based mini project in 1st year of Master of Computer Application in Computer Science and Engineering from Madhav Institute of Technology and Science, Gwalior.

Dr. Anshu Chaturvedi
(Professor)
Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

DECLARATION

I hereby declare that the work being presented in this project report, for the fulfillment of partial requirement for the skills based mini project in 1st year of Master of Computer Application in Computer Science and Engineering at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Dr. Anshu Chaturvedi** (Professor), MITS Gwalior.

I declare that I have not submitted the matter embodied in this report anywhere else.

Vinayak Choubey
0901CA211068
2021-2023 Year,
Master of Computer Application,
Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE , GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science,** for allowing me to continue my disciplinary project. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Department of Computer Science and Engineering,** for allowing me to explore this project. I humbly thank **Dr. Manish Dixit,** Professor and Head, Department of Computer Science and Engineering, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty coordinator. I am grateful to the guidance of **Dr. Anshu Chaturvedi** (Professor), Computer Science and Engineering, for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

Vinayak Choubey
0901CA211068
2021-2023 Year,
Master of Computer Application,
Computer Science and Engineering

ABSTRACT

This project aims to bring the fun and simplicity of the snake game with some new features. It will include computer controlled intelligent opponents whose aim will be to challenge the human players. It will also have the multiplayer feature that will allow more than one player to play the game over a network.

This project explores a new dimension in the traditional snake game to make it more interesting and challenging. The simplicity of this game makes it an ideal candidate for a minor project as we can focus on advanced topics like multiplayer functionality and implementation of computer controlled intelligent opponents.

CONTENTS

1. INTRODUCTION

This project creates the Snake Game, based on a GUI application which is implemented in java using a swing. In this game, the player controls a snake. The objective is to eat as many apples as possible. Each time the snake eats an apple its body grows. The number of apples eaten will be calculated in the form of a score. The snake must avoid the walls and its own body, if the player fails in avoiding the collision then the game will be over. The game is sometimes called *Nibbles.*

Playing games is fun and exciting. It gives us relief from stress and unwinds from our stressful work. Many of us spend our free time or others that use most of their time in playing and exploring new games. Today, with the raped development of technology we have, games that are rising up together with it.

Nowadays with technology we have many games that are developed for computers specifically for windows. With the high technology equipped with these computer games become robust and attract many people to buy or have this gadget for them to experience what's inside it which makes it a trend for the new generation of gadget.

2. OBJECTIVE

Snake game is one of the most popular arcade games of all time. In this game, the main objective of the player is to catch the maximum number of fruits without hitting the wall or itself. Creating a snake game can be taken as a challenge while learning Python or Pygame. It is one of the best beginner – friendly projects that every novice programmer should take as a challenge. Learning to build a video game is kinda interesting and fun learning.

3. CODE

3.1. Main.java

```
package com.company;

public class Main {

  public static void main(String[] args) {
      new Game();
  }
}
```

3.2. Game.java

```
package com.company;

import javax.swing.*;

public class Game extends JFrame {

  public Game() {
      this.add(new Graphics());
      this.setTitle("Snake Game");
      this.pack();
      this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
      this.setResizable(false);
      this.setVisible(true);
      this.setLocationRelativeTo(null);
  }
}
```

3.3. Food.java

```
package com.company;
import java.util.Random;
public class Food {

    private final int posX;
    private final int posY;

    public Food() {
        posX = generatePos(Graphics.WIDTH);
        posY = generatePos(Graphics.HEIGHT);
    }

    private int generatePos(int size) {
        Random random = new Random();
        return random.nextInt(size / Graphics.TICK_SIZE) * Graphics.TICK_SIZE;
    }

    public int getPosX() {
        return posX;
    }

    public int getPosY() {
        return posY;
    }
}
```

3.4. Graphics.java

```
package com.company;

import javax.swing.*;
```

```
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;

class Graphics extends JPanel implements ActionListener {

  static final int WIDTH = 800;
  static final int HEIGHT = 800;
  static final int TICK_SIZE = 50;
  static final int BOARD_SIZE = (WIDTH * HEIGHT) / (TICK_SIZE * TICK_SIZE);

  final Font font = new Font("TimesRoman", Font.BOLD, 30);

  int[] snakePosX = new int[BOARD_SIZE];
  int[] snakePosY = new int[BOARD_SIZE];
  int snakeLength;

  Food food;
  int foodEaten;

  char direction = 'R';
  boolean isMoving = false;
  final Timer timer = new Timer(150, this);

  public Graphics() {
      this.setPreferredSize(new Dimension(WIDTH, HEIGHT));
      this.setBackground(Color.WHITE);
      this.setFocusable(true);
      this.addKeyListener(new KeyAdapter() {
          @Override
          public void keyPressed(KeyEvent e) {
```

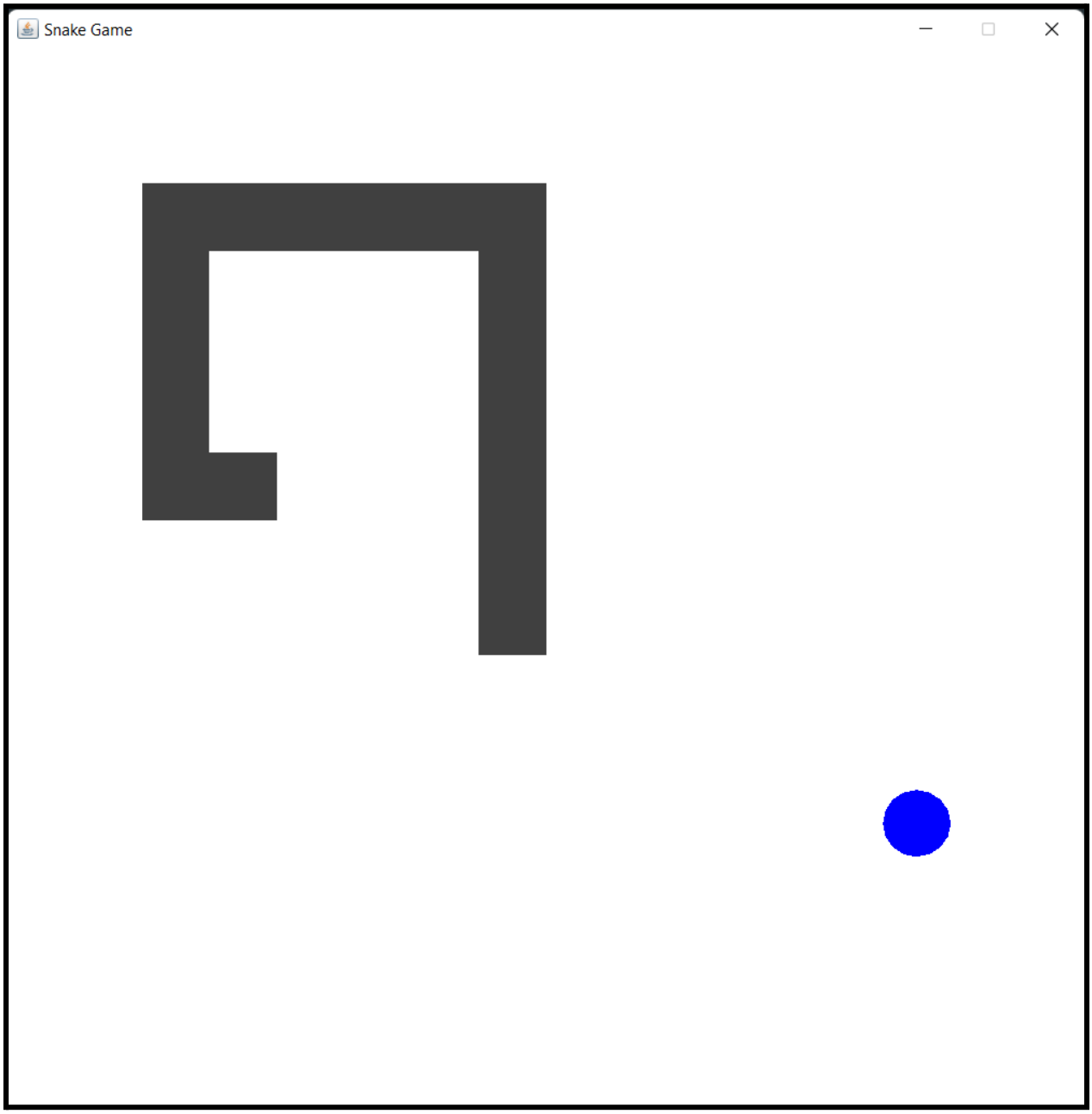
```
            if (isMoving) {
                switch (e.getKeyCode()) {
                    case KeyEvent.VK_LEFT:
                        if (direction != 'R') {
                            direction = 'L';
                        }
                        break;
                    case KeyEvent.VK_RIGHT:
                        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;
                }
            } else {
                start();
            }
        }
    });

    start();
}

protected void start() {
```

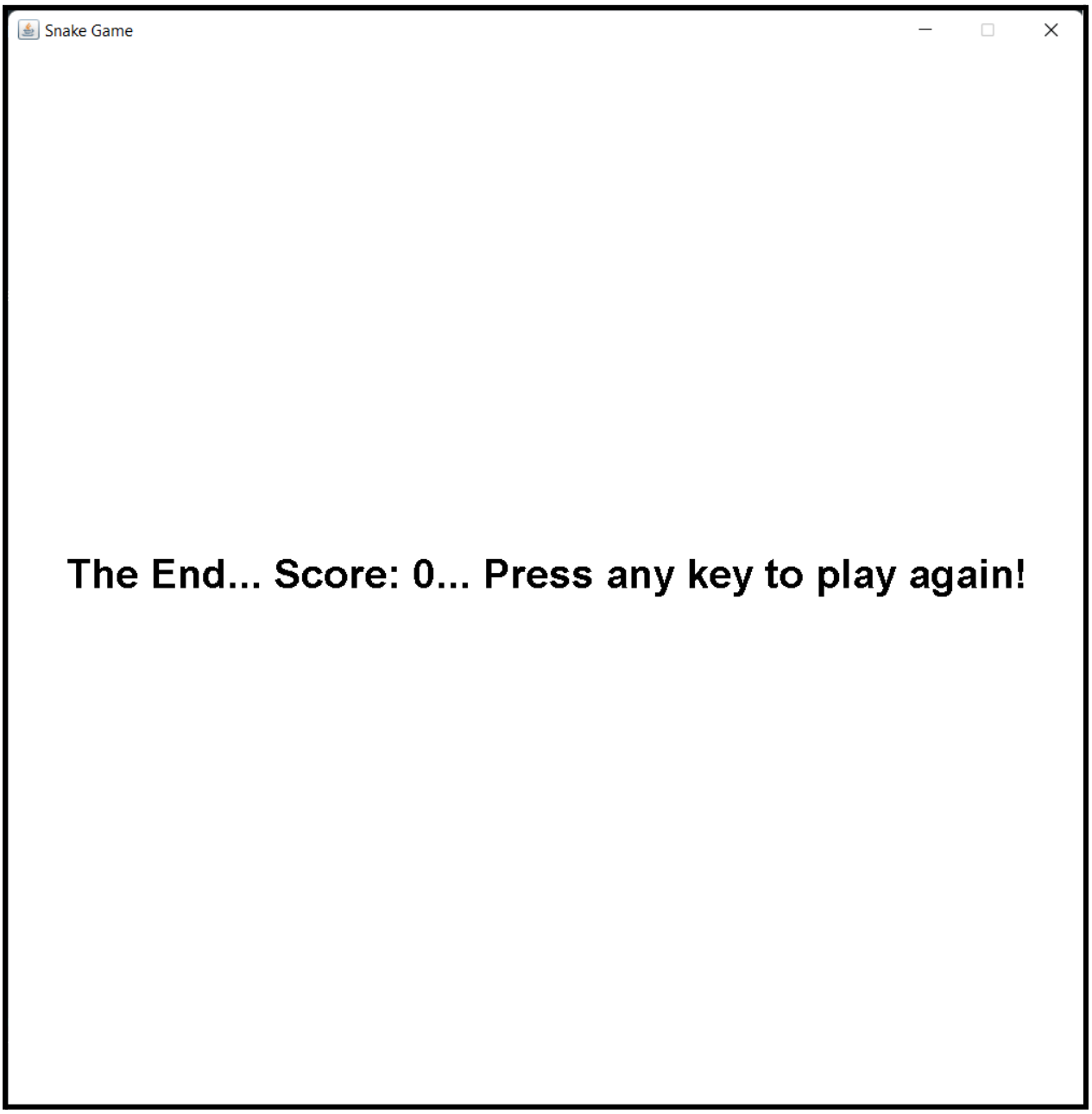
```
        snakePosX = new int[BOARD_SIZE];
        snakePosY = new int[BOARD_SIZE];
        snakeLength = 5;
        foodEaten = 0;
        direction = 'R';
        isMoving = true;
        spawnFood();
        timer.start();
    }

    @Override
    protected void paintComponent(java.awt.Graphics g) {
        super.paintComponent(g);

        if (isMoving) {
            g.setColor(Color.BLUE);
            g.fillOval(food.getPosX(), food.getPosY(), TICK_SIZE, TICK_SIZE);

            g.setColor(Color.DARK_GRAY);
            for (int i = 0; i < snakeLength; i++) {
                g.fillRect(snakePosX[i], snakePosY[i], TICK_SIZE, TICK_SIZE);
            }
        } else {
            String scoreText = String.format("The End... Score: %d... Press any key to play
again!", foodEaten);
            g.setColor(Color.BLACK);
            g.setFont(font);
            g.drawString(scoreText, (WIDTH -
getFontMetrics(g.getFont()).stringWidth(scoreText)) / 2, HEIGHT / 2);
        }
    }

    protected void move() {
```

```
        for (int i = snakeLength; i > 0; i--) {
            snakePosX[i] = snakePosX[i-1];
            snakePosY[i] = snakePosY[i-1];
        }

        switch (direction) {
            case 'U' -> snakePosY[0] -= TICK_SIZE;
            case 'D' -> snakePosY[0] += TICK_SIZE;
            case 'L' -> snakePosX[0] -= TICK_SIZE;
            case 'R' -> snakePosX[0] += TICK_SIZE;
        }
    }

    protected void spawnFood() {
        food = new Food();
    }

    protected void eatFood() {
        if ((snakePosX[0] == food.getPosX()) && (snakePosY[0] == food.getPosY())) {
            snakeLength++;
            foodEaten++;
            spawnFood();
        }
    }

    protected void collisionTest() {
        for (int i = snakeLength; i > 0; i--) {
            if ((snakePosX[0] == snakePosX[i]) && (snakePosY[0] == snakePosY[i])) {
                isMoving = false;
                break;
            }
        }
```

```
        if (snakePosX[0] < 0 || snakePosX[0] > WIDTH - TICK_SIZE || snakePosY[0] < 0 ||
snakePosY[0] > HEIGHT - TICK_SIZE) {
            isMoving = false;
        }

        if (!isMoving) {
            timer.stop();
        }
    }

    @Override
    public void actionPerformed(ActionEvent e) {
        if (isMoving) {
            move();
            collisionTest();
            eatFood();
        }

        repaint();
    }
}
```

4. OUTPUT




Snake Game
The End... Score: 0... Press any key to play again!

5. CONCLUSION

The project in JAVA programming of Snake Game is a simple console application with very simple graphics. In this project, you can play the popular "Snake Game" just like you played it elsewhere. You have to use the up, down, right, or left arrows to move the snake.

Foods are provided at the several coordinates of the screen for the snake to eat. Every time the snake eats the food. its length will be increased by one element along with the score. • It isn't the world's greatest game, but it does give you an idea of what you can achieve with relatively simple python programming, and perhaps the basis by which to extend the principles and create more interesting games on your own.

6. BIBLIOGRAPHY

1. **http://www.google.com**
2. **https://www.tutorialspoint.com**
3. **https://www.javapoint.com**
4. **https://stackoverflow.com/questions/6190599/snake_game**