A PROJECT REPORT ON

"SNAKE GAME USING JAVA"

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENT FOR DEGREE OF

MASTERS OF COMPUTER APPLICATIONS
(SEMESTER I)

<u>SUBMITTED BY</u>

SOURABH SABLANIA Roll: 354

<u>SUBMITTED TO</u>

SAVITRIBAI PHULE PUNE UNIVERSITY

MASTERS OF COMPUTER APPLICATION

DR.D.Y.PATIL INSTITUTE OF MANAGEMENT & RESEARCH

2020-2022

INDEX

INTRODUCTION

The project is concerned with the development of the software program, in this case, a Snake Game using JAVA.

JAVA AWT components and Swing components were mostly used in the development of this project.

This game can be played using a keyboard. It consists of three levels which can either be accessed by reaching the required score or by using the dedicated buttons. It also stores the high score and if a new high score is made it stores it as the new high score and high score is displayed.

This project documentation includes user documentation which tells users how to use the software product and system documentation which is principally intended for further development and understanding.

REQUIREMENTS

- <u>HARDWARE REQUIREMENTS:</u>

  - INTEL CORE 2 DUO 2.0GHz Processor.
  - 512MB RAM or higher.
  - 2GB HARD DRIVE.
  - KEYBOARD.
  - MOUSE.

- <u>SOFTWARE REQUIREMENTS:</u>

  - JDK 1.8.0 or higher.
  - JRE 8 or higher.
  - Netbeans IDE.
  - Windows 7 or higher.

FEASIBILITY STUDY

- ECONOMIC FEASIBILITY:

This study is carried out to check the economic impact that the project will have on the organization. The project is economically feasible as it didn't cost much and most of the resources used were available for free.

Eg:

The IDE used, i.e., NETBEANS is a free IDE developed by APACHE SOFTWARE FOUNDATION and ORACLE CORP.

- TECHNICAL FEASIBILITY:

This study is carried out to check the technical feasibility, that is, the technical requirements of the project. The described project does not have a high demand on the available technical resource and only minimal or null changes are required for the implementing this system.

- OPERATIONAL FEASIBILITY:

The aspect of study is to check the level of acceptance of the program by the user. This includes the process of training the user to be able to use it efficiently. The proposed project is easy to use and efficient instructions are given to the user about the game.

OBJECTIVES

I. Developing a Snake game.
II. Creating multiple levels of the game.
III. Increase difficulty according to levels.
IV. Increase the length as the snake eats the food.
V. Store the score.
VI. Display Game Over when the snake bites itself or bumps into the obstacles.
VII. Take the user to the next level upon reaching a specific score.

SCOPE

I. Single Player.
II. Three Levels.
III. Different Difficulty levels for all levels.
IV. Can be played by using a Keyboard.

PROJECT DESCRIPTION

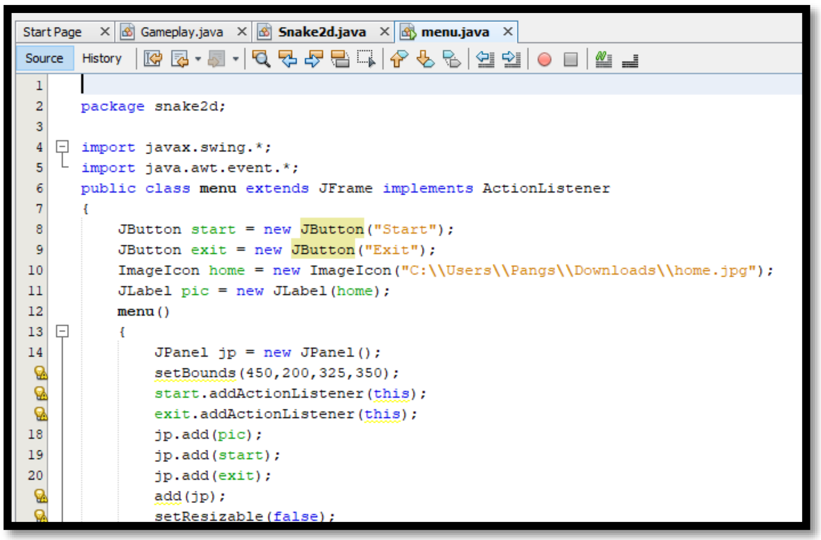
I. SYSTEM ARCHITECTURE:

System architecture defines the working methodology of the game and shows the components, their relationships and how they evolve to make the game work.

a) GAME COMPONENTS:

i. Menu or Main Class:

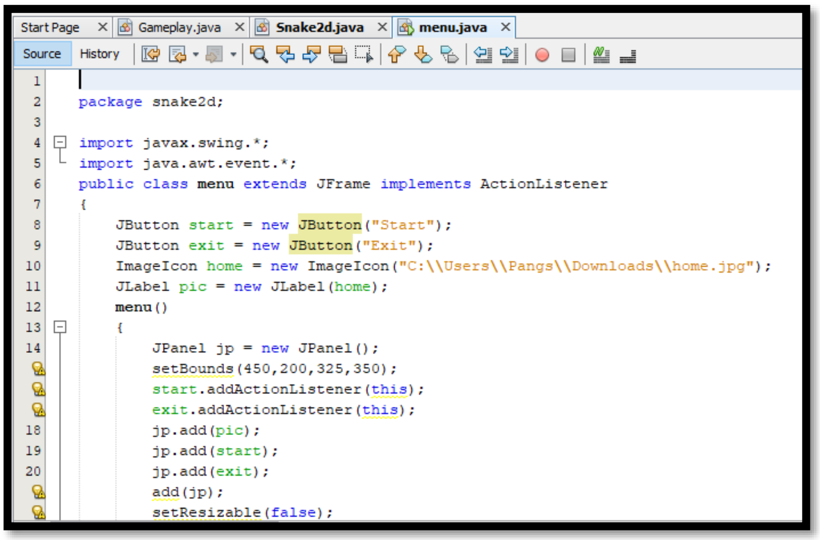
The game starts to execute from this class. It displays two buttons, i.e, start and exit. Upon Clicking start it instantiates an object of other class which draws the Frame for gameplay and set its attributes and If exit is clicked it exits the program.



```

package snake2d;

import javax.swing.*;
import java.awt.event.*;
public class menu extends JFrame implements ActionListener
{
    JButton start = new JButton("Start");
    JButton exit = new JButton("Exit");
    ImageIcon home = new ImageIcon("C:\\Users\\Pangs\\Downloads\\home.jpg");
    JLabel pic = new JLabel(home);
    menu()
    {
        JPanel jp = new JPanel();
        setBounds(450,200,325,350);
        start.addActionListener(this);
        exit.addActionListener(this);
        jp.add(pic);
        jp.add(start);
        jp.add(exit);
        add(jp);
        setResizable(false);
```

ii. Intermediate class:

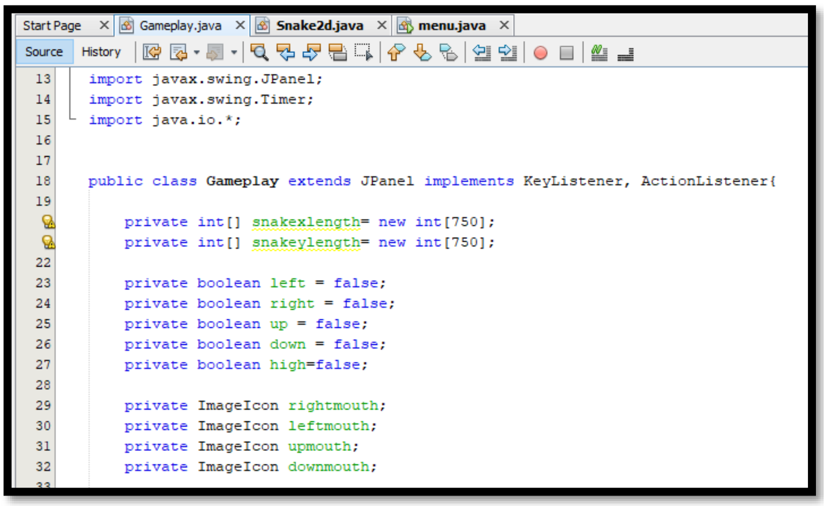
It draws the frame in which the game is to be played or displayed and sets its attributes like Visible,Resizable,color,size,etc. It also instantiates the object of the gameplay class in which the main code will be found and implements the ActionListener and KeyListener.



```java

package snake2d;

import javax.swing.*;
import java.awt.event.*;
public class menu extends JFrame implements ActionListener
{
    JButton start = new JButton("Start");
    JButton exit = new JButton("Exit");
    ImageIcon home = new ImageIcon("C:\\Users\\Pangs\\Downloads\\home.jpg");
    JLabel pic = new JLabel(home);
    menu()
    {
        JPanel jp = new JPanel();
        setBounds(450,200,325,350);
        start.addActionListener(this);
        exit.addActionListener(this);
        jp.add(pic);
        jp.add(start);
        jp.add(exit);
        add(jp);
        setResizable(false);
```

iii. Gameplay Class:

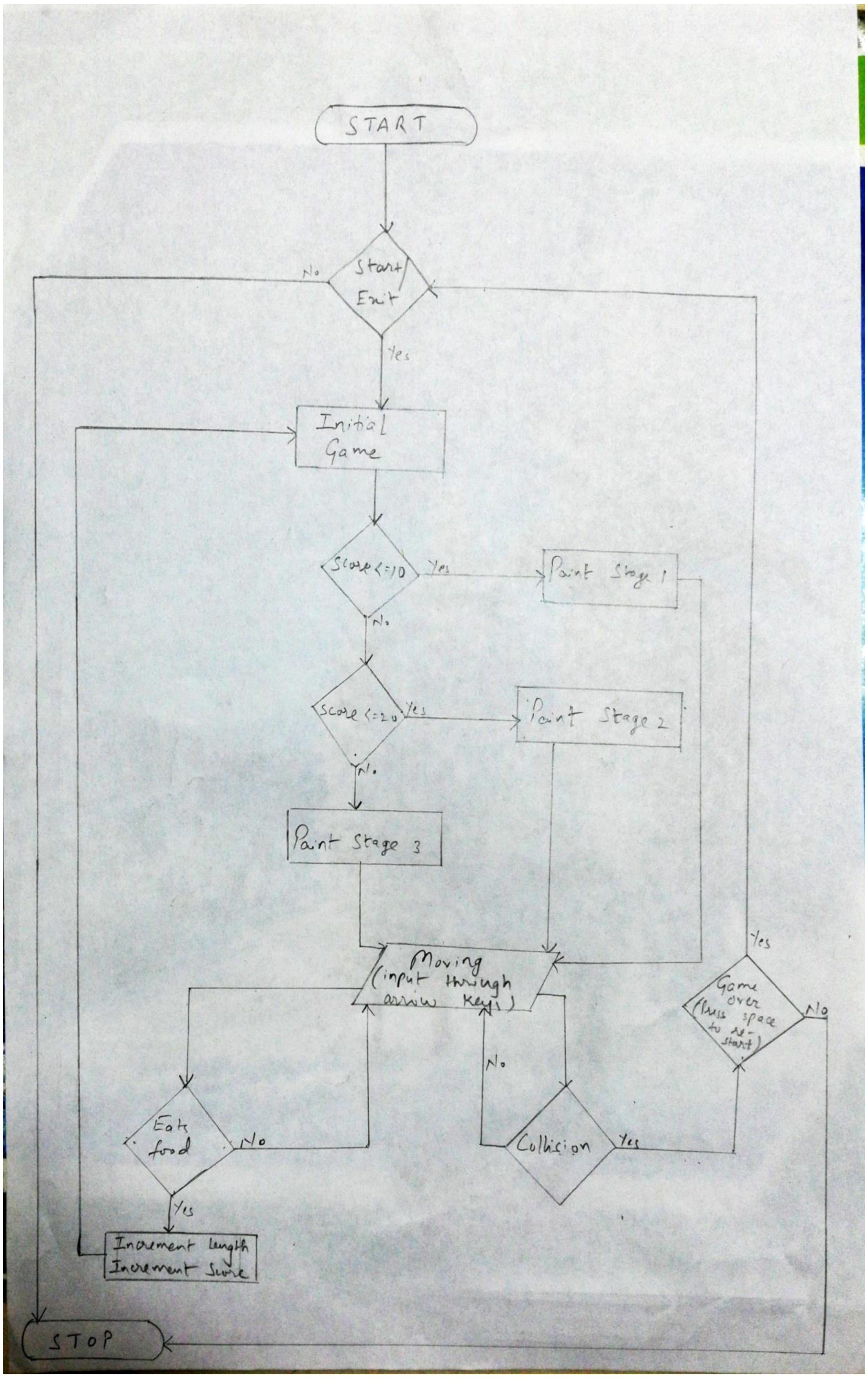
This class contains the code for the gameplay and implements many components like ActionListener,etc. It also displays the snake and the food and also controls the movement of the snake and randomization of the coordinates of the food. It also controls the levels.



```java
import javax.swing.JPanel;
import javax.swing.Timer;
import java.io.*;


public class Gameplay extends JPanel implements KeyListener, ActionListener{

    private int[] snakexlength= new int[750];
    private int[] snakeylength= new int[750];

    private boolean left = false;
    private boolean right = false;
    private boolean up = false;
    private boolean down = false;
    private boolean high=false;

    private ImageIcon rightmouth;
    private ImageIcon leftmouth;
    private ImageIcon upmouth;
    private ImageIcon downmouth;
```

b) GAME ARCHITECTURE:

The below diagram explains the game architecture or working of this game.



II. BUILDING THE GAME:

It consisted of various phases like planning, designing, coding, etc.

We first started planning, figured out the scope and objectives and designed the game architecture. Then, after designing the implementation and coding was the most challenging part of the process. Testing of the game was done frequently and also after every change, minor or major to ensure that everything is working as it was supposed to.

This game has been fully developed using JAVA programming language by using Java AWT and Swing components.

The development phases are:

- ✓ Creating a main menu for the game.
- ✓ Creating a frame in which the game is to be played.
- ✓ Drawing the snake and the food on the frame.
- ✓ Changing level according to the score.
- ✓ Creating obstacles or walls as the levels go higher.
- ✓ Maintaining and displaying score and length and high score.
- ✓ Displaying Game Over if the snake bites itself or the walls.
- ✓ Testing and Fixing Errors (repetitive).

TESTING

I. UNIT TESTING:

Every module or functionality was tested to ensure that each of them are working properly. This helps minimize risks of errors in further testing as it can become a real problem if we can’t figure out the source at later stages.

We tested the movement of the snake properly before moving on to test further.

II. REGRESSION TESTING:

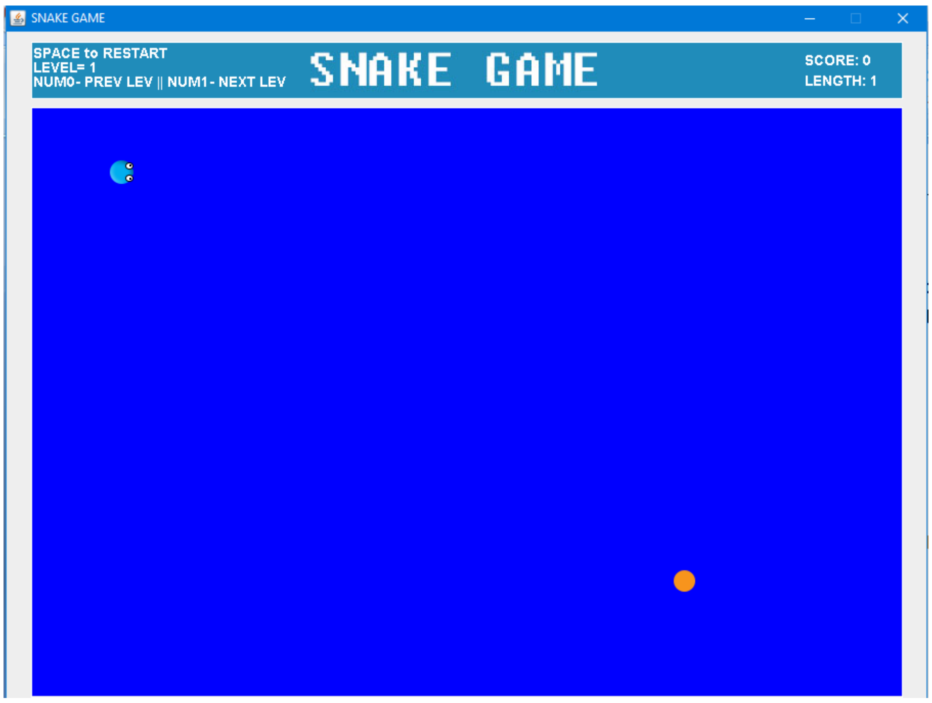
The game was tested after every change made to it, either minor or major. We re-checked to analyse the working of the previous functionalities of the game works fine and that new changes have not introduced any new errors or vulnerabilities. It saves time as it detects the errors early in development.

We tested the movement of the snake first, but after implementing the food part, it was re-tested again from the start, then again after the obstacles or walls were introduced.

III. GAMEPLAY TESTING:

Play testing was done to ensure that the game is built properly as per the requirements and is well structured. In most cases, this is done by some select users.

We tested the gameplay so as to ensure that the game meets the specified requirements and its UI is properly implemented.



SOURCE CODE

I. MENU or MAIN CLASS:

```
package snake2d;
import javax.swing.*;
import java.awt.event.*;
public class menu extends JFrame implements ActionListener
{
    JButton start = new JButton("Start");
    JButton exit = new JButton("Exit");
    ImageIcon home = new ImageIcon("C:\\Users\\Pangs\\Downloads\\home.jpg");
    JLabel pic = new JLabel(home);
    menu()
    {
        JPanel jp = new JPanel();
        setBounds(450,200,325,350);
        start.addActionListener(this);
        exit.addActionListener(this);
        jp.add(pic);
        jp.add(start);
        jp.add(exit);
        add(jp);
        setResizable(false);
        setVisible(true);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    }
    public void actionPerformed(ActionEvent evt)
    {
        Object source = evt.getSource();
        if(source==exit)
        {
            System.exit(0);
        }

        if(source==start)

        {

            Snake2d d = new Snake2d();

            setVisible(false);

        }

    }

    public static void main(String args[])

    {

        menu ob = new menu();

    }

}
```

II. INTERMEDIATE CLASS:

```
package snake2d;

import java.awt.Color;
import javax.swing.JFrame;

public class Snake2d {


    Snake2d(){

        JFrame obj = new JFrame("SNAKE GAME");
        Gameplay gameplay = new Gameplay();
        obj.setBounds(10,10,905,700);
        obj.setBackground(Color.DARK_GRAY);
        obj.setResizable(false);
        obj.setVisible(true);
        obj.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        obj.add(gameplay);

    }

}
```

III. GAMEPLAY CLASS:

```
package snake2d;



import java.awt.Color;
import java.awt.Font;

import java.awt.Graphics;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyEvent;

import java.awt.event.KeyListener;

import java.util.Random;

import javax.swing.ImageIcon;

import javax.swing.JPanel;

import javax.swing.Timer;

import java.io.*;



public class Gameplay extends JPanel implements KeyListener, ActionListener{


    private int[] snakexlength= new int[750];

    private int[] snakeylength= new int[750];


    private boolean left = false;

    private boolean right = false;

    private boolean up = false;

    private boolean down = false;

    private boolean high=false;


    private ImageIcon rightmouth;

    private ImageIcon leftmouth;

    private ImageIcon upmouth;

    private ImageIcon downmouth;


    private Timer timer;

    private int delay = 150;

    private ImageIcon SnakeImage;
```

```
		private ImageIcon obs;

		private ImageIcon obs2;


		private int lengthofsnake=1;

		private int moves=0;


		private int score=0;

		private int level=1;


		private int
enemyxpos[]={75,100,125,150,175,200,225,250,275,300,325,350,375,400,450,475,500,525,550,575,600,62
5,650,675,700,725,750,775,800};

		private int
enemyypos[]={125,150,175,200,225,250,275,300,350,375,400,425,450,475,500,525,550,575};


		private ImageIcon enemyimage;


		private Random random = new Random();


		private int xpos = random.nextInt(29);

		private int ypos = random.nextInt(18);


		private ImageIcon titleImage;

		public Gameplay()

		{

		    addKeyListener(this);

		    setFocusable(true);

		    setFocusTraversalKeysEnabled(false);

		    timer = new Timer(delay, this);

		    timer.start();

		}


		public void paint(Graphics g)

		{
```

```
                    if(moves==0)
                    {
                        snakexlength[2]=50;
                        snakexlength[1]=75;
                        snakexlength[0]=100;

                        snakeylength[2]=125;
                        snakeylength[1]=125;
                        snakeylength[0]=125;
                    }



                    //title image border
                    g.setColor(Color.white);
                    g.drawRect(24, 10, 851, 55);

                    //title image
                    titleImage = new
ImageIcon("C:\\Users\\Pangs\\Downloads\\Compressed\\assets\\snake.jpg");
                    titleImage.paintIcon(this, g, 25, 11);

                    //draw border for gameplay
                    g.setColor(Color.white);
                    g.drawRect(24, 74, 851, 577);

                    //draw background gameplay
                    g.setColor(Color.blue);
                    g.fillRect(25, 75, 850, 575);

                    //draw score
                    g.setColor(Color.white);
                    g.setFont(new Font("Helvetica",Font.BOLD,14));
```

```
g.drawString("SCORE: "+score, 780, 33);


g.drawString("LENGTH: "+lengthofsnake, 780, 53);




g.drawString("SPACE to RESTART",26,26);




g.drawString("LEVEL= "+level,26,40);

g.drawString("NUM0- PREV LEV || NUM1- NEXT LEV",26,54);


if(level>=2)

{

obs = new ImageIcon("C:\\Users\\Pangs\\Desktop\\aa.png");

obs.paintIcon(this, g, 425, 400);

obs.paintIcon(this, g, 425, 450);

obs.paintIcon(this, g, 425, 225);

obs.paintIcon(this, g, 425, 175);

obs2=new ImageIcon("C:\\Users\\Pangs\\Desktop\\aa2.png");

obs2.paintIcon(this, g, 525, 325);

obs2.paintIcon(this, g, 575, 325);

obs2.paintIcon(this, g, 250, 325);

obs2.paintIcon(this, g, 200, 325);

}


if(level==3)

{

    obs = new ImageIcon("C:\\Users\\Pangs\\Desktop\\aa3.png");

    obs.paintIcon(this, g, 50, 100);

    obs.paintIcon(this, g, 50, 150);

    obs.paintIcon(this, g, 50, 225);

    obs.paintIcon(this, g, 50, 300);
    obs.paintIcon(this, g, 50, 375);
    obs.paintIcon(this, g, 50, 450);
    obs.paintIcon(this, g, 50, 525);
    obs.paintIcon(this, g, 50, 550);
    obs.paintIcon(this, g, 825, 100);
    obs.paintIcon(this, g, 825, 150);
    obs.paintIcon(this, g, 825, 225);
    obs.paintIcon(this, g, 825, 300);
    obs.paintIcon(this, g, 825, 375);
    obs.paintIcon(this, g, 825, 450);
    obs.paintIcon(this, g, 825, 525);
    obs.paintIcon(this, g, 825, 550);
    obs2 = new ImageIcon("C:\\Users\\Pangs\\Desktop\\aa4.png");
    obs2.paintIcon(this, g, 50, 100);
    obs2.paintIcon(this, g, 125, 100);
    obs2.paintIcon(this, g, 200, 100);
    obs2.paintIcon(this, g, 275, 100);
    obs2.paintIcon(this, g, 350, 100);
    obs2.paintIcon(this, g, 425, 100);
    obs2.paintIcon(this, g, 500, 100);
    obs2.paintIcon(this, g, 575, 100);
    obs2.paintIcon(this, g, 650, 100);
    obs2.paintIcon(this, g, 725, 100);
    obs2.paintIcon(this, g, 750, 100);
    obs2.paintIcon(this, g, 50, 600);
    obs2.paintIcon(this, g, 125, 600);
    obs2.paintIcon(this, g, 200, 600);
    obs2.paintIcon(this, g, 275, 600);
    obs2.paintIcon(this, g, 350, 600);
    obs2.paintIcon(this, g, 425, 600);
    obs2.paintIcon(this, g, 500, 600);
    obs2.paintIcon(this, g, 575, 600);
```

```
                 obs2.paintIcon(this, g, 650, 600);

                 obs2.paintIcon(this, g, 725, 600);

                 obs2.paintIcon(this, g, 750, 600);

              }


              //draw snake

              rightmouth = new
ImageIcon("C:\\Users\\Pangs\\Downloads\\Compressed\\assets\\rightmouth.png");

              rightmouth.paintIcon(this, g, snakexlength[0], snakeylength[0]);


              for(int a=0;a<lengthofsnake;a++)

              {

                  if(a==0 && right)

                  {

                      rightmouth = new
ImageIcon("C:\\Users\\Pangs\\Downloads\\Compressed\\assets\\rightmouth.png");

                      rightmouth.paintIcon(this, g, snakexlength[a], snakeylength[a]);

                  }

                  if(a==0 && left)

                  {

                      leftmouth = new
ImageIcon("C:\\Users\\Pangs\\Downloads\\Compressed\\assets\\leftmouth.png");

                      leftmouth.paintIcon(this, g, snakexlength[a], snakeylength[a]);

                  }

                  if(a==0 && down)

                  {

                      downmouth = new
ImageIcon("C:\\Users\\Pangs\\Downloads\\Compressed\\assets\\downmouth.png");

                      downmouth.paintIcon(this, g, snakexlength[a], snakeylength[a]);

                  }

                  if(a==0 && up)

                  {

                      upmouth = new
ImageIcon("C:\\Users\\Pangs\\Downloads\\Compressed\\assets\\upmouth.png");
```

```
                upmouth.paintIcon(this, g, snakexlength[a], snakeylength[a]);

            }

            if(a!=0)

            {

                SnakeImage = new
ImageIcon("C:\\Users\\Pangs\\Downloads\\Compressed\\assets\\snakeimage.png");

                SnakeImage.paintIcon(this, g, snakexlength[a], snakeylength[a]);

            }

        }


        enemyimage = new
ImageIcon("C:\\Users\\Pangs\\Downloads\\Compressed\\assets\\enemy.png");


        if(enemyxpos[xpos]==snakexlength[0]&&enemyypos[ypos]==snakeylength[0])

        {

            score++;

            lengthofsnake++;

            xpos=random.nextInt(29);

            ypos=random.nextInt(18);

        }


        enemyimage.paintIcon(this, g, enemyxpos[xpos], enemyypos[ypos]);


       if(score==10&&level==1)

        {

            level++;

            lengthofsnake=1;

            moves=0;

            timer.stop();

            g.setColor(Color.white);

            g.setFont(new Font("Candy Round BTN",Font.TRUETYPE_FONT,80));

            g.drawString("LEVEL 2",350,300);
```

```
    g.setColor(Color.white);

    g.setFont(new Font("Candy Round BTN",Font.BOLD,40));

    g.drawString("PRESS N to CONTINUE",300,350);


    if(moves==0)

{

    snakexlength[2]=50;

    snakexlength[1]=75;

    snakexlength[0]=100;


    snakeylength[2]=100;

    snakeylength[1]=100;

    snakeylength[0]=100;

}

}

if(score==20&&level==2)

{

    level++;

    lengthofsnake=1;

    moves=0;

    timer.stop();

    g.setColor(Color.white);

    g.setFont(new Font("Candy Round BTN",Font.TRUETYPE_FONT,80));

    g.drawString("LEVEL 3",350,300);


    g.setColor(Color.white);

    g.setFont(new Font("Candy Round BTN",Font.BOLD,40));

    g.drawString("PRESS N to CONTINUE",300,350);


    if(moves==0)

{

    snakexlength[2]=75;
```

```
            snakexlength[1]=100;

            snakexlength[0]=125;


            snakeylength[2]=125;

            snakeylength[1]=125;

            snakeylength[0]=125;

        }

        }


        //middle walls

        if(level>1)

        {

if(snakexlength[0]==425&&snakeylength[0]==400||snakexlength[0]==425&&snakeylength[0]==425||sna
kexlength[0]==425&&snakeylength[0]==450||snakexlength[0]==425&&snakeylength[0]==475||snakexle
ngth[0]==425&&snakeylength[0]==500||snakexlength[0]==425&&snakeylength[0]==175||snakexlength[0
]==425&&snakeylength[0]==200||snakexlength[0]==425&&snakeylength[0]==225||snakexlength[0]==42
5&&snakeylength[0]==250||snakexlength[0]==425&&snakeylength[0]==275||snakexlength[0]==525&&s
nakeylength[0]==325||snakexlength[0]==550&&snakeylength[0]==325||snakexlength[0]==575&&snakey
length[0]==325||snakexlength[0]==600&&snakeylength[0]==325||snakexlength[0]==625&&snakeylengt
h[0]==325||snakexlength[0]==200&&snakeylength[0]==325||snakexlength[0]==225&&snakeylength[0]==
325||snakexlength[0]==250&&snakeylength[0]==325||snakexlength[0]==275&&snakeylength[0]==325||s
nakexlength[0]==300&&snakeylength[0]==325)

            {


                right=false;

                left=false;

                up=false;

                down=false;


                obs=new ImageIcon("C:\\Users\\Pangs\\Desktop\\cr.png");

                obs.paintIcon(this, g, snakexlength[0], snakeylength[0]);

                g.setColor(Color.white);

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

                g.drawString("GAME OVER",300,300);


                g.setColor(Color.white);
                g.setFont(new Font("arial",Font.BOLD,20));

                g.drawString("SPACE to RESTART",350,340);



                timer.stop();


            }

        }

        //lev3 walls

        if(level==3)

        {

            //upper wall

if(snakexlength[0]==50&&snakeylength[0]==100||snakexlength[0]==75&&snakeylength[0]==100||snakex
length[0]==100&&snakeylength[0]==100||snakexlength[0]==125&&snakeylength[0]==100||snakexlength[
0]==150&&snakeylength[0]==100||snakexlength[0]==175&&snakeylength[0]==100||snakexlength[0]==200
&&snakeylength[0]==100||snakexlength[0]==225&&snakeylength[0]==100||snakexlength[0]==250&&sna
keylength[0]==100||snakexlength[0]==275&&snakeylength[0]==100||snakexlength[0]==300&&snakeylen
gth[0]==100||snakexlength[0]==325&&snakeylength[0]==100||snakexlength[0]==350&&snakeylength[0]=
=100||snakexlength[0]==375&&snakeylength[0]==100||snakexlength[0]==400&&snakeylength[0]==100||s
nakexlength[0]==425&&snakeylength[0]==100||snakexlength[0]==450&&snakeylength[0]==100||snakex
length[0]==475&&snakeylength[0]==100||snakexlength[0]==500&&snakeylength[0]==100||snakexlength
[0]==525&&snakeylength[0]==100||snakexlength[0]==550&&snakeylength[0]==100||snakexlength[0]==5
75&&snakeylength[0]==100||snakexlength[0]==600&&snakeylength[0]==100||snakexlength[0]==625&&s
nakeylength[0]==100||snakexlength[0]==650&&snakeylength[0]==100||snakexlength[0]==675&&snakeyl
ength[0]==100||snakexlength[0]==700&&snakeylength[0]==100||snakexlength[0]==725&&snakeylength[
0]==100||snakexlength[0]==750&&snakeylength[0]==100||snakexlength[0]==775&&snakeylength[0]==10
0||snakexlength[0]==800&&snakeylength[0]==100||snakexlength[0]==825&&snakeylength[0]==100)

            {

                try

                {

                    file();

                }

                catch(IOException e)

                {

                    e.printStackTrace();

                }


                right=false;
```

```
                left=false;

                up=false;

                down=false;


                g.setColor(Color.white);

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

                g.drawString("GAME OVER",300,300);


                g.setColor(Color.white);

                g.setFont(new Font("arial",Font.BOLD,20));

                g.drawString("SPACE to RESTART",350,340);

                // print high score

                if(high)

                {

                    g.setColor(Color.green);

                    g.setFont(new Font("arial",Font.ITALIC,30));

                    g.drawString("Congratulations !! High Score",260,390);

                }

                timer.stop();

            }

            //left wall

if(snakexlength[0]==50&&snakeylength[0]==125||snakexlength[0]==50&&snakeylength[0]==150||snakex
length[0]==50&&snakeylength[0]==175||snakexlength[0]==50&&snakeylength[0]==200||snakexlength[0
]==50&&snakeylength[0]==225||snakexlength[0]==50&&snakeylength[0]==250||snakexlength[0]==50&&
snakeylength[0]==275||snakexlength[0]==50&&snakeylength[0]==300||snakexlength[0]==50&&snakeyl
ength[0]==325||snakexlength[0]==50&&snakeylength[0]==350||snakexlength[0]==50&&snakeylength[0]
==375||snakexlength[0]==50&&snakeylength[0]==400||snakexlength[0]==50&&snakeylength[0]==425||s
nakexlength[0]==50&&snakeylength[0]==450||snakexlength[0]==50&&snakeylength[0]==475||snakexle
ngth[0]==50&&snakeylength[0]==500||snakexlength[0]==50&&snakeylength[0]==550||snakexlength[0]=
=50&&snakeylength[0]==575)

            {

                 try

                {

                    file();

                }
```

```
                        catch(IOException e)
                        {
                            e.printStackTrace();
                        }


                        right=false;
                        left=false;
                        up=false;
                        down=false;


                        g.setColor(Color.white);
                        g.setFont(new Font("arial",Font.BOLD,50));
                        g.drawString("GAME OVER",300,300);


                        g.setColor(Color.white);
                        g.setFont(new Font("arial",Font.BOLD,20));
                        g.drawString("SPACE to RESTART",350,340);
                        // print high score
                        if(high)
                        {
                            g.setColor(Color.green);
                            g.setFont(new Font("arial",Font.ITALIC,30));
                            g.drawString("Congratulations !! High Score",260,390);
                        }
                        timer.stop();
                    }

                    //right wall

if(snakexlength[0]==825&&snakeylength[0]==125||snakexlength[0]==825&&snakeylength[0]==150||snak
exlength[0]==825&&snakeylength[0]==175||snakexlength[0]==825&&snakeylength[0]==200||snakexlen
gth[0]==825&&snakeylength[0]==225||snakexlength[0]==825&&snakeylength[0]==250||snakexlength[0]
==825&&snakeylength[0]==275||snakexlength[0]==825&&snakeylength[0]==300||snakexlength[0]==825
&&snakeylength[0]==325||snakexlength[0]==825&&snakeylength[0]==350||snakexlength[0]==825&&sn
akeylength[0]==375||snakexlength[0]==825&&snakeylength[0]==400||snakexlength[0]==825&&snakeyl
```

```
ength[0]==425||snakexlength[0]==825&&snakeylength[0]==450||snakexlength[0]==825&&snakeylength
[0]==475||snakexlength[0]==825&&snakeylength[0]==500||snakexlength[0]==825&&snakeylength[0]==5
50||snakexlength[0]==825&&snakeylength[0]==575)

                    {

                      try

                       {

                          file();

                       }

                       catch(IOException e)

                       {

                          e.printStackTrace();

                       }


                       right=false;

                       left=false;

                       up=false;

                       down=false;


                       g.setColor(Color.white);

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

                       g.drawString("GAME OVER",300,300);


                       g.setColor(Color.white);

                       g.setFont(new Font("arial",Font.BOLD,20));

                       g.drawString("SPACE to RESTART",350,340);

                       // print high score

                       if(high)

                       {

                          g.setColor(Color.green);

                          g.setFont(new Font("arial",Font.ITALIC,30));

                          g.drawString("Congratulations !! High Score",260,390);

                       }

                       timer.stop();
                    }


                    //lower wall

if(snakexlength[0]==50&&snakeylength[0]==600||snakexlength[0]==75&&snakeylength[0]==600||snake
xlength[0]==100&&snakeylength[0]==600||snakexlength[0]==125&&snakeylength[0]==600||snakexlengt
h[0]==150&&snakeylength[0]==600||snakexlength[0]==175&&snakeylength[0]==600||snakexlength[0]==
200&&snakeylength[0]==600||snakexlength[0]==225&&snakeylength[0]==600||snakexlength[0]==250&
&snakeylength[0]==600||snakexlength[0]==275&&snakeylength[0]==600||snakexlength[0]==300&&sna
keylength[0]==600||snakexlength[0]==325&&snakeylength[0]==600||snakexlength[0]==350&&snakeyle
ngth[0]==600||snakexlength[0]==375&&snakeylength[0]==600||snakexlength[0]==400&&snakeylength[
0]==600||snakexlength[0]==425&&snakeylength[0]==600||snakexlength[0]==450&&snakeylength[0]==6
00||snakexlength[0]==475&&snakeylength[0]==600||snakexlength[0]==500&&snakeylength[0]==600||s
nakexlength[0]==525&&snakeylength[0]==600||snakexlength[0]==550&&snakeylength[0]==600||snake
xlength[0]==575&&snakeylength[0]==600||snakexlength[0]==600&&snakeylength[0]==600||snakexleng
th[0]==625&&snakeylength[0]==600||snakexlength[0]==650&&snakeylength[0]==600||snakexlength[0]=
=675&&snakeylength[0]==600||snakexlength[0]==700&&snakeylength[0]==600||snakexlength[0]==725&
&snakeylength[0]==600||snakexlength[0]==750&&snakeylength[0]==600||snakexlength[0]==775&&sna
keylength[0]==600||snakexlength[0]==800&&snakeylength[0]==600||snakexlength[0]==825&&snakeyle
ngth[0]==600)
                    {
                        try
                       {
                          file();
                       }
                       catch(IOException e)
                       {
                          e.printStackTrace();
                       }


                       right=false;
                       left=false;
                       up=false;
                       down=false;
                       g.setColor(Color.white);
                       g.setFont(new Font("arial",Font.BOLD,50));
                       g.drawString("GAME OVER",300,300);


                       g.setColor(Color.white);
```

```
            g.setFont(new Font("arial",Font.BOLD,20));
            g.drawString("SPACE to RESTART",350,340);
            // print high score
            if(high)
            {
                g.setColor(Color.green);
                g.setFont(new Font("arial",Font.ITALIC,30));
                g.drawString("Congratulations !! High Score",260,390);
            }
            timer.stop();
        }
    }

    for(int b=1; b<lengthofsnake;b++)
    {
        if(snakexlength[b]==snakexlength[0]&&snakeylength[b]==snakeylength[0])
        {
             try
            {
                file();
            }
            catch(IOException e)
            {
                e.printStackTrace();
            }
            right=false;
            left=false;
            up=false;
            down=false;

            g.setColor(Color.white);
            g.setFont(new Font("arial",Font.BOLD,50));
            g.drawString("GAME OVER",300,300);


            g.setColor(Color.white);
            g.setFont(new Font("arial",Font.BOLD,20));
            g.drawString("SPACE to RESTART",350,340);
            // print high score
            if(high)
            {
                g.setColor(Color.green);
                g.setFont(new Font("arial",Font.ITALIC,30));
                g.drawString("Congratulations !! High Score",260,390);
            }
            timer.stop();

        }

    }





    g.dispose();
}
@Override
public void keyTyped(KeyEvent e) {

}

@Override
public void keyPressed(KeyEvent e) {

     if(e.getKeyCode() == KeyEvent.VK_NUMPAD0)
```

```
{
    if(level>1)
    {
    level--;
    lengthofsnake=1;
    moves=0;
    score=0;
    repaint();
    timer.start();
    }
}
  if(e.getKeyCode() == KeyEvent.VK_NUMPAD1)
{
    if(level<3)
    {
    level++;
    lengthofsnake=1;
    moves=0;
    score=0;
    repaint();
    timer.start();
    }
}
 if(e.getKeyCode() == KeyEvent.VK_SPACE)
{
    moves=0;
    score=0;
    level=1;
    lengthofsnake=1;
    high=false;
    repaint();
    timer.start();
```

```
}

if(e.getKeyCode()==KeyEvent.VK_N)
{
    repaint();
    timer.start();
}

if(e.getKeyCode() == KeyEvent.VK_RIGHT)
{
    moves++;
    right=true;
    if(!left)
    {
        right=true;
    }
    else
    {
        right=false;
        left=true;
    }
    up = false;
    down = false;
}
if(e.getKeyCode()== KeyEvent.VK_LEFT)
{
    moves++;
    left=true;
    if(!right)
    {
        left=true;
    }
    else
    {
        left=false;
        right=true;
    }
    up = false;
    down = false;
}
if(e.getKeyCode()== KeyEvent.VK_UP)
{
    moves++;
    up=true;
    if(!down)
    {
        up=true;
    }
    else
    {
        up=false;
        down=true;
    }
    left = false;
    right = false;
}if(e.getKeyCode()== KeyEvent.VK_DOWN)
{
    moves++;
    down=true;
    if(!up)
    {
        down=true;
    }
    else
```

```
            {
                down=false;
                up=true;
            }
            left = false;
            right = false;
        }

    }

    @Override
    public void keyReleased(KeyEvent e) {

    }

    @Override
    public void actionPerformed(ActionEvent e) {
        timer.start();
        if(right)
        {
            for(int r=lengthofsnake-1;r>=0;r--)
            {
                snakeylength[r+1]=snakeylength[r];
            }
            for(int r=lengthofsnake-1;r>=0;r--)
            {
                if(r==0)
                {
                    snakexlength[r]=snakexlength[r]+25;
                }
                else
                {
```

```
                snakexlength[r]=snakexlength[r-1];
            }
            if(snakexlength[r]>850)
            {
                snakexlength[r]=25;
            }
        }
        repaint();
    }
    if(left)
    {
        for(int r=lengthofsnake-1;r>=0;r--)
        {
            snakeylength[r+1]=snakeylength[r];
        }
        for(int r=lengthofsnake-1;r>=0;r--)
        {
            if(r==0)
            {
                snakexlength[r]=snakexlength[r]-25;
            }
            else
            {
                snakexlength[r]=snakexlength[r-1];
            }
            if(snakexlength[r]<25)
            {
                snakexlength[r]=850;
            }
        }
        repaint();
    }
```

```
if(up)
{
    for(int r=lengthofsnake-1;r>=0;r--)
    {
        snakexlength[r+1]=snakexlength[r];
    }
    for(int r=lengthofsnake-1;r>=0;r--)
    {
        if(r==0)
        {
            snakeylength[r]=snakeylength[r]-25;
        }
        else
        {
            snakeylength[r]=snakeylength[r-1];
        }
        if(snakeylength[r]<75)
        {
            snakeylength[r]=625;
        }
    }
    repaint();
}
if(down)
{
    for(int r=lengthofsnake-1;r>=0;r--)
    {
        snakexlength[r+1]=snakexlength[r];
    }
    for(int r=lengthofsnake-1;r>=0;r--)
    {
        if(r==0)
```

```
            {
                snakeylength[r]=snakeylength[r]+25;
            }
            else
            {
                snakeylength[r]=snakeylength[r-1];
            }
            if(snakeylength[r]>625)
            {
                snakeylength[r]=75;
            }
        }
        repaint();
    }
}

public void file()throws IOException
{
    String str="";
    int max=0,num=0;
    // File Handling
    FileReader file_r = new FileReader("C:\\Users\\Pangs\\Desktop\\scores.txt");
    BufferedReader ob = new BufferedReader(file_r);
    while((str=ob.readLine())!=null)
    {
        num = Integer.parseInt(str);
        if(num > max)
            max=num;
    }
    if(score > max)
        high=true;
    ob.close();
```

```
            FileWriter file_w = new FileWriter("C:\\Users\\Pangs\\Desktop\\scores.txt",true);
            PrintWriter pw = new PrintWriter(file_w);
            if(score > 0)
              pw.println(score);
            pw.close();
         }
      }
```

USER MANUAL

I. GAME REQUIREMENTS:

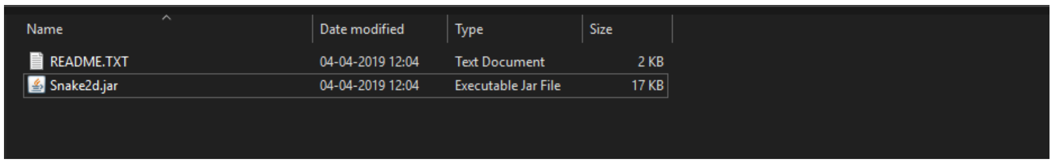
a. Intel Core 2 Duo 2Ghz processor.
b. 512MB RAM.
c. 1GB of Hard Disk Space.
d. Mouse.
e. Keyboard.
f. A display device.

II. GAME CONTROLS:

UP arrow- go up	DOWN arrow-go down
LEFT arrow- go left	RIGHT arrow-go right
SPACE- RESTART	N- Go to next level after level complete
NUM0- Prev. level	NUM1- Next Level.

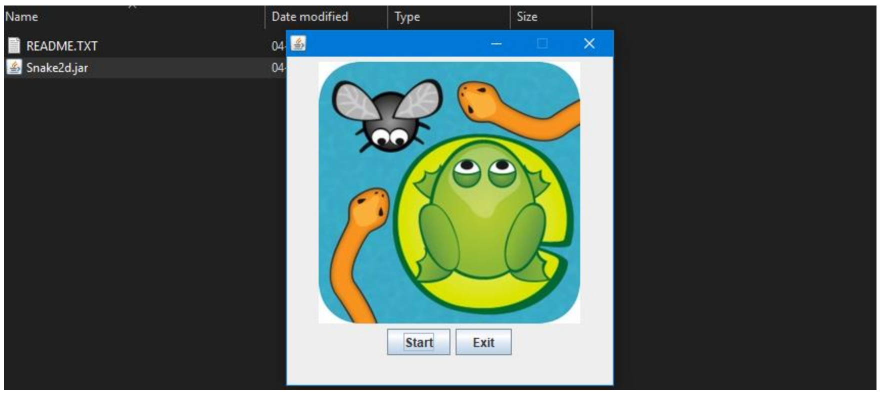
III. STEPS TO START THE GAME:

a. Go to the directory which contains the game files (.jar).

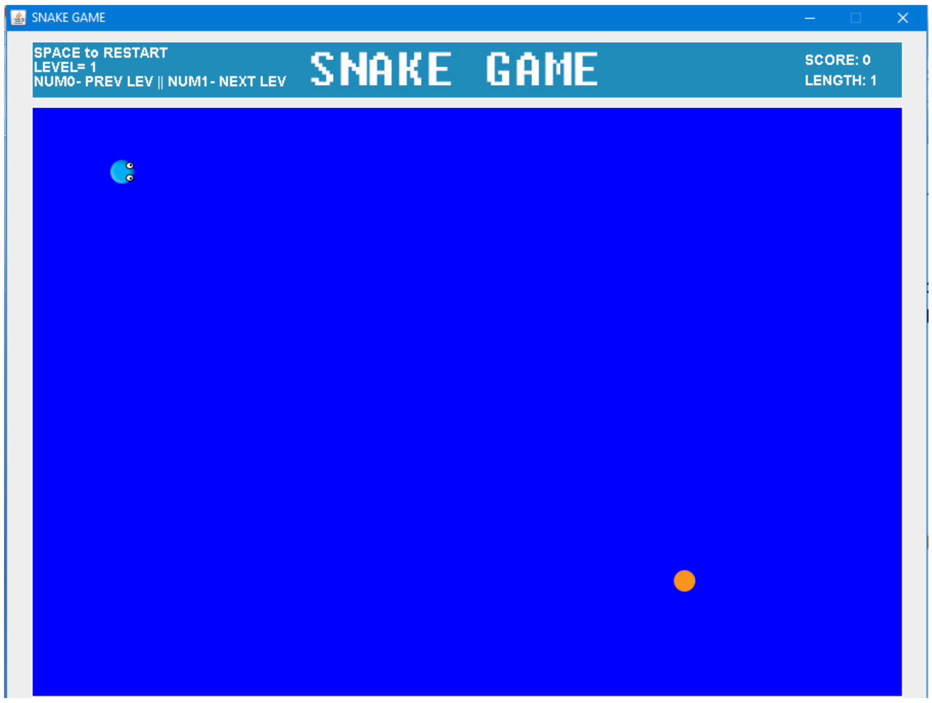


Name	Date modified	Type	Size
README.TXT	04-04-2019 12:04	Text Document	2 KB
Snake2d.jar	04-04-2019 12:04	Executable Jar File	17 KB

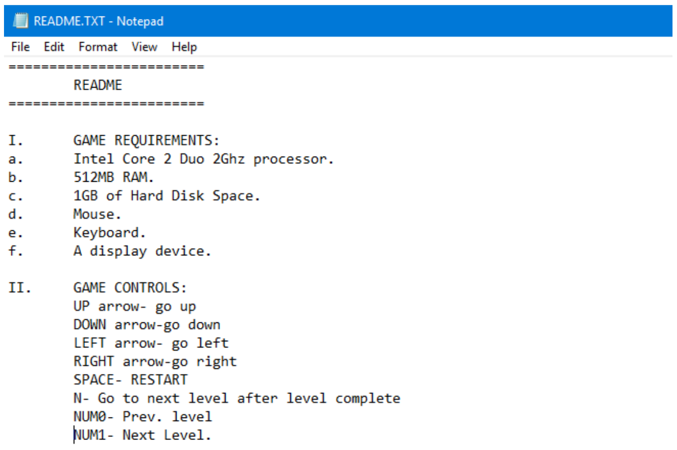
b. Double-click on the Snake2d.jar file to open it. The main menu will be displayed.



c. Click on Start to start the Game and Exit to quit the game.



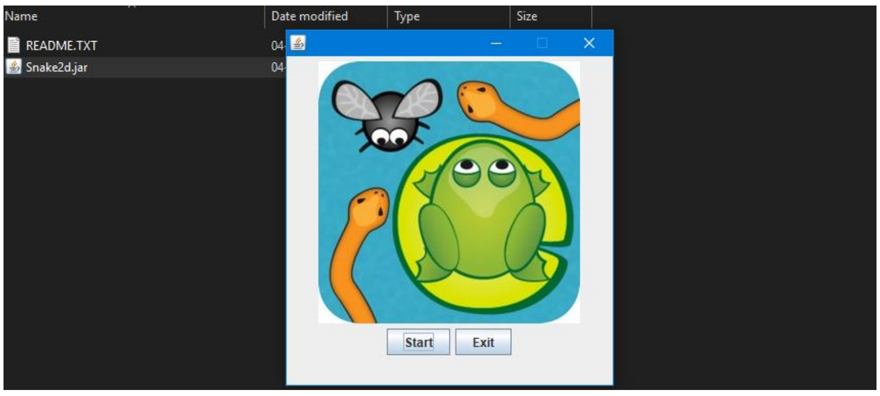
d. Use arrow keys to move the snake. For other key bindings, refer the readme.txt file inside the game directory.



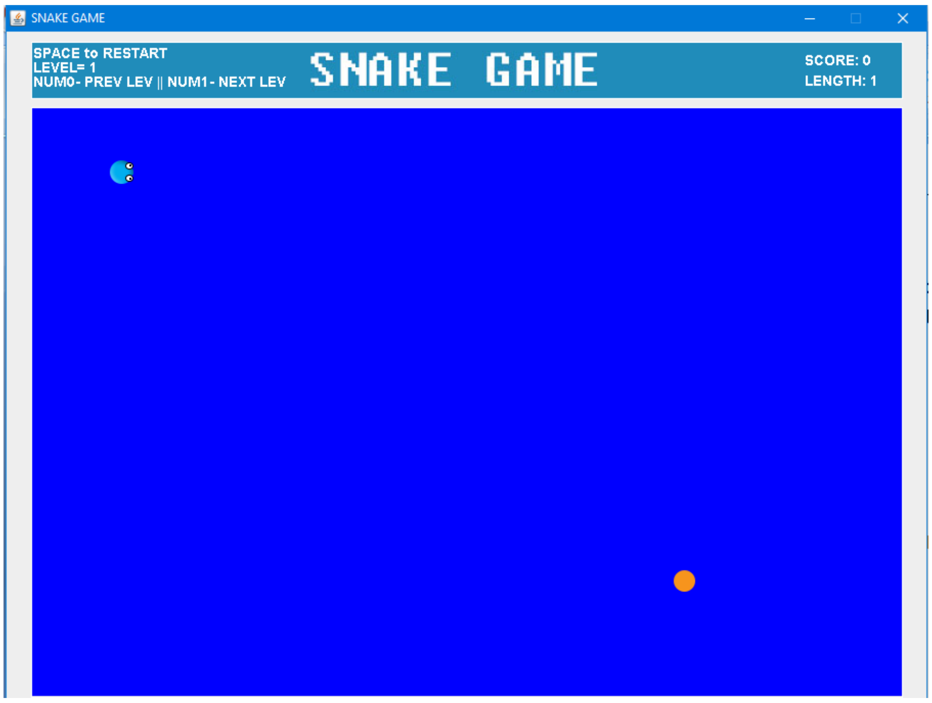
```
========================
        README
========================

I.      GAME REQUIREMENTS:
a.      Intel Core 2 Duo 2Ghz processor.
b.      512MB RAM.
c.      1GB of Hard Disk Space.
d.      Mouse.
e.      Keyboard.
f.      A display device.

II.     GAME CONTROLS:
        UP arrow- go up
        DOWN arrow-go down
        LEFT arrow- go left
        RIGHT arrow-go right
        SPACE- RESTART
        N- Go to next level after level complete
        NUM0- Prev. level
        NUM1- Next Level.
```

GAME USER INTERFACES

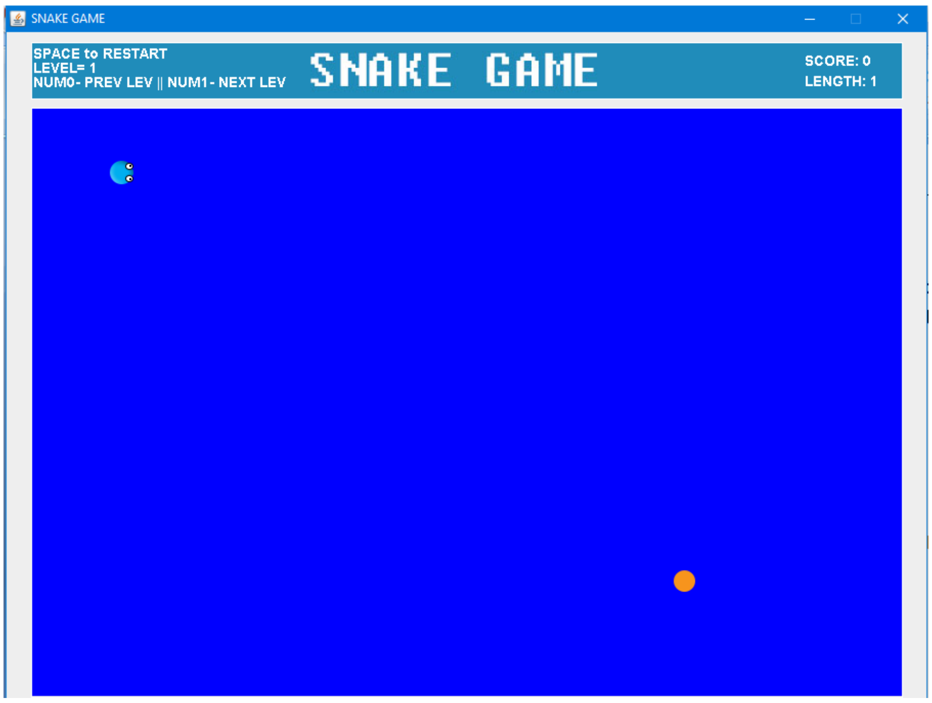
I. Menu Screen:



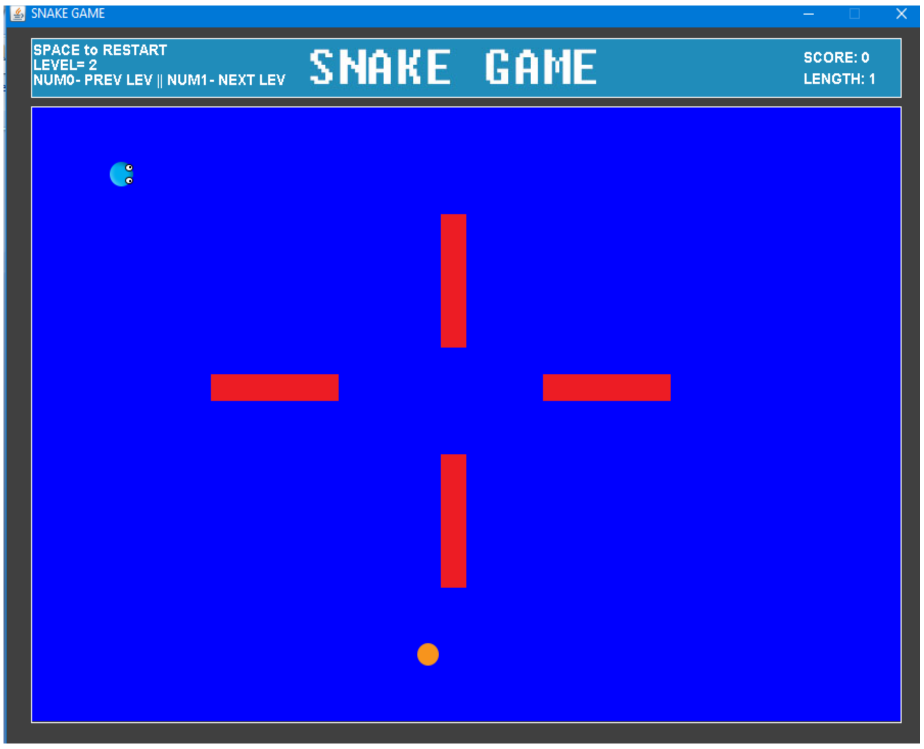
II. Gameplay Screen:



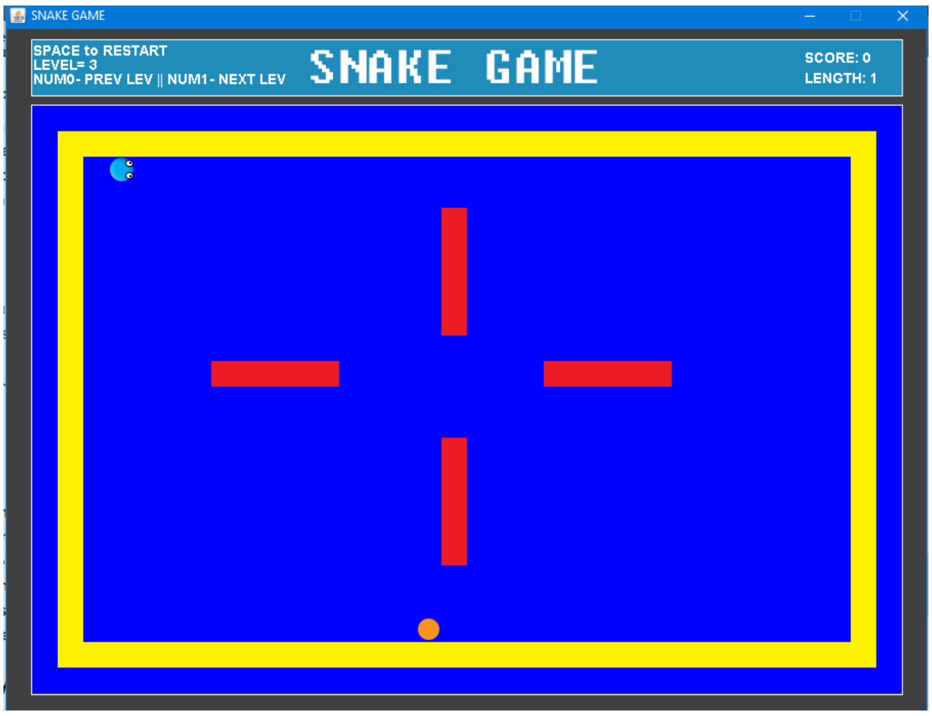
III. LEVELS:



LEVEL -1

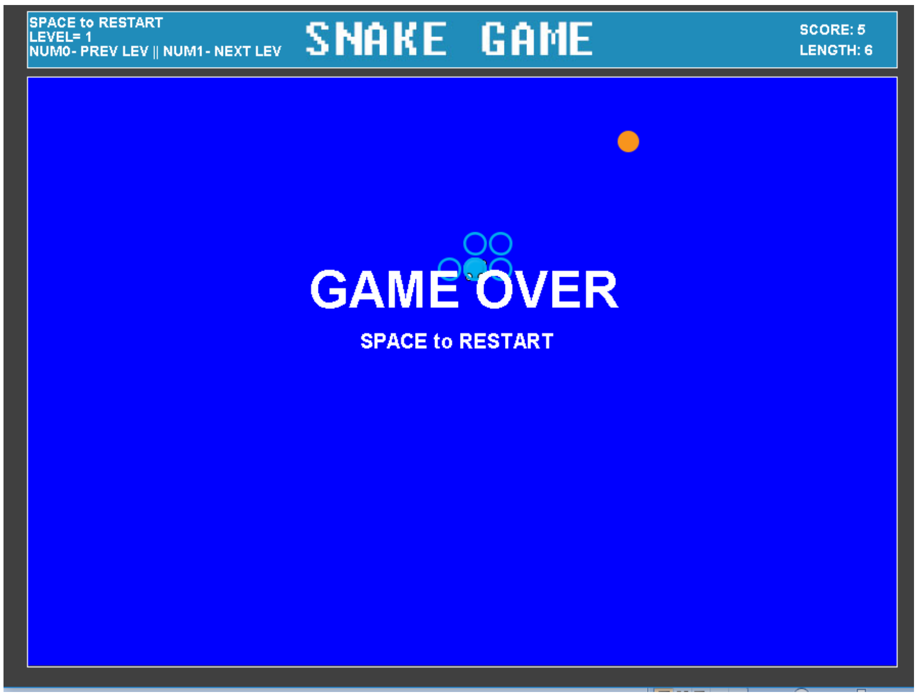


LEVEL -2

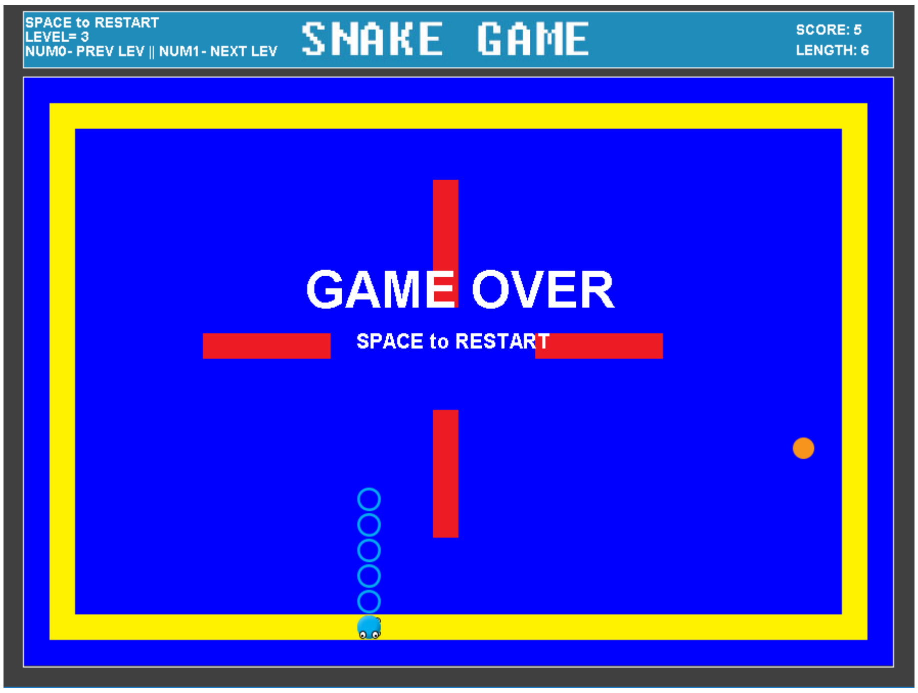


LEVEL -3

IV. GAME OVER:

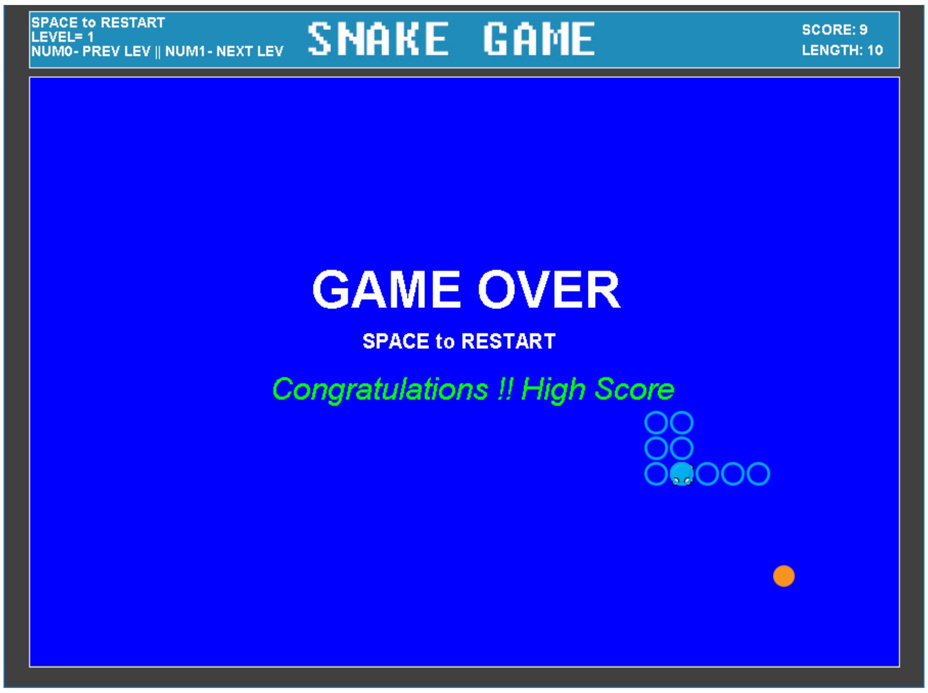


Snake bites itself.

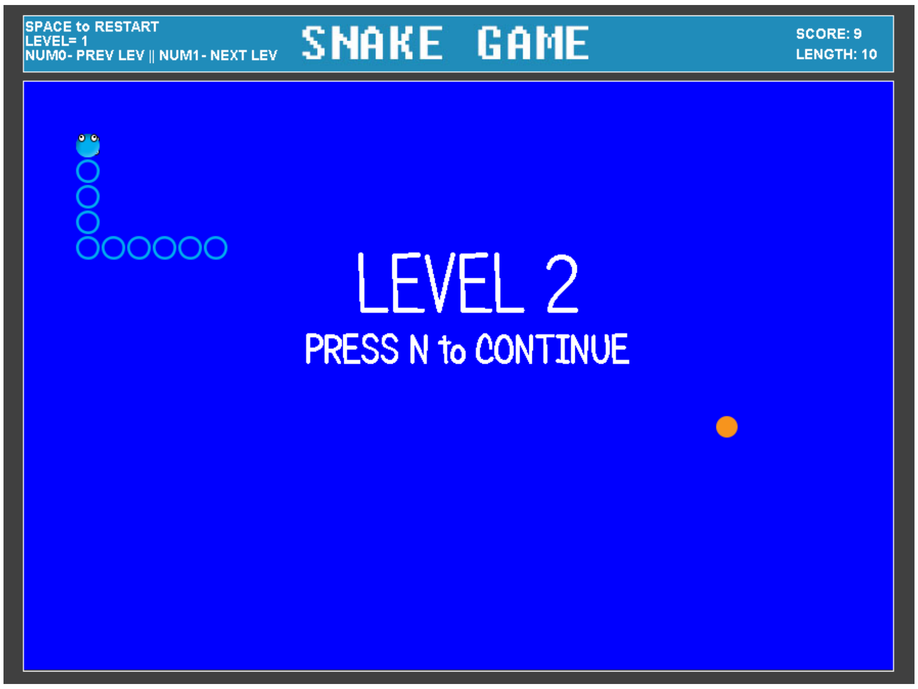


Snake bumps the wall.

V. HIGH SCORE:



VI. WHEN SCORE IS REACHED:



CONCLUSION

After thorough investigation of the game functionalities and the code, etc, we can finally conclude that the project is fully functional and is as per all the requirements and hence can be concluded as complete.

Through this project, we have learned about the aspects of game development and it also helped improve our programming knowledge.

FUTURE SCOPE AND ENHANCEMENTS

I. More levels can be added with increased difficulty.

II. Option for map customization by user (custom maps).

III. More functionalities like recording the time spent to complete a level and total time spent on the game, and missions can be implemented like, "Complete a level within 60secs", etc.

BIBLIOGRAPHY

WEBSITES:

1. Google [https://google.com]
2. Stack Overflow [https://stackoverflow.com]
3. Geeks For Geeks [https://www.geeksforgeeks.org]