A Mini Project Report

On

Snake and ladder

Submitted in partial fulfillment of requirements for the Course
CSE18R272 - JAVA PROGRAMMING

Bachelor's of Technology

In

Computer Science and Engineering

Submitted By

T SHASHIPREETHAM REDDY

9918004115

S SAI GNANESHWAR REDDY

9918004113

Under the guidance of

Dr. R. RAMALAKSHMI

(Associate Professor)

Department of Computer Science and Engineering

Kalasalingam Academy of Research and Education

Anand Nagar, Krishnankoil-626126

April 2020

ABSTRACT

The idea of our project is to design "Snake and Ladder" game. A snake and ladder is a simple board game which can be played between two players. The two players race their token from start to end, to finish the game by rolling the dice. The board is played on numbered grid squares which are filled with ladders and snakes on selected boxes. This is a game of chance and depends on the person's luck. The size of the grid is 10x10 and varies from board to board which will affect the duration play. Player who reaches the 100th box first will be the winner of the game.

DECLARATION

I hereby declare that the work presented in this report entitled "**Snake and ladder**", in partial fulfilment of the requirements for the award of the degree **Bachelors in Technology** and submitted in **Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education,Krishnankoil** (Affiliated to **Deemed to be University**) is an authentic record of my own work carried out during the period from **Jan 2020** under the guidance of **DR. R. RAMA LAKSHMI** (ASSOCIATE PROFESSOR).

The work reported in this has not been submitted by me for the award of any other degree of this or any other institute.

T SHASHIPREETHAM REDDY
9918004115
S SAI GNANESHWAR REDDY
9918004113

ACKNOWLEDGEMENT

First and foremost, I wish to thank the **Almighty God** for his grace and benediction tocomplete this Project work successfully. I would like to convey my special thanks from the bottom of my heart to my dear **Parents** and affectionate **Family members** for their honest support for the completion of this Project work.
I express deep sense of gratitude to "Kalvivallal" Thiru. **T. Kalasalingam** B.com., Founder Chairman, "Ilayavallal" **Dr.K.Sridharan** Ph.D., Chancellor, **Dr.S.ShasiAnand**, Ph.D., Vice President (Academic) , **Mr.S.ArjunKalasalingam** M.S., Vice President (Administration) , **Dr.R.Nagaraj** Vice-Chancellor, **Dr.V.Vasudevan** Ph.D., Registrar , **Dr.P.Deepalakshmi** Ph.D., Dean (School of Computing) . And also a special thanks to **Dr. A. FRANCIS SAVIOUR DEVARAJ.** Head Department of CSE, Kalasalingam Academy of Research and Education forgranting the permission and providing necessary facilities to carry out Project work.
I would like to express my special appreciation and profound thanks to my enthusiastic Project Supervisor **Dr.R.Ramalakshmi** Ph.D, Associate Professor at Kalasalingam Academy of Research and Education [KARE] for her inspiring guidance, constant encouragement with my work during all stages. I am extremely glad that I had a chance to do my Project under my Guide, who truly practices and appreciates deep thinking. And during the most difficult times when writing this report, he gave me the moral support and the freedom I needed to move on

T SHASHIPREETHAM REDDY
9918004115
S SAI GNANESHWAR REDDY
9918004113

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

INTRODUCTION

The idea of our project is to design "Snake and Ladder" game. A snake and ladder game is a simple board game which can be played between two players. The players race their token from start to end, to finish the game by rolling their dice. The board is played on numbered grid squares which are filled with ladders and snakes on selected boxes. This is a game of chance and depends on the person's luck. The size of the grid is 10x10 and varies from board to board which will affect the duration play. In this multi-player snakes and ladders game, a minimum of two players and maximum of 4 players can play this game. In the beginning of the game, the player's tokens are placed in the start area. Every player will be given a chance and they have to go about it turn wise. The buttons are moved according to the number that is appeared on rolled dice. Once the player's position is at the bottom of the ladder then he can climb it and reach to the top of the ladder. If the player encounters a snake he has to retrace his steps back by coming to the square where the tail of the snake ends.one who reaches the finish point first is declared as the winner

Chapter 2

HISTORY

1.Snakes and Ladders originated in India as a game based on morality called Vaikuntapaali or Parama-pada Sopanam (the ladder to salvation).
The game was played widely in ancient India by the name
2.The game was played widely in ancient India by the name of Moksha Patamu, the earliest known Jain version Gyanbazi dating back to 16th century.
3.Moksha Patamu was perhaps invented by Hindu spiritual
4.Moksha Patamu was perhaps invented by Hindu spiritual teachers to teach children about the effects of good deeds as opposed to bad deeds. The ladders

representedvirtues such asgenerosity, faith, humility, etc., and the snakes represented vices such as anger, theft, etc.

5.Variations of the game made it to England during the British Raj, with one appearing under the name Snakes and Ladders around 1892, which was then patented

Chapter 3

REQUIREMENT SPECIFICATIONS

2.1-TECHNOLOGIES USED:

–>Java AWT

–> Java Swing

-> Java

2.2–SOFTWARE ENGINEERING PARADIGM:

->JAVA AWT:

Java AWT (Abstract Window Toolkit) is an API to develop GUI or window-based applications in java. Java AWT (Abstract Window Toolkit) is an API to develop GUI or window-based applications in java. Java AWT components are platform-dependent i.e. components are displayed according to the view of operating system. AWT is heavy-weight i.e. its components are using the resources of OS. The java.awt package provides classes for AWT api such as Text Field, Label, Text Area, Radio Button, Checkbox, Choice, List etc. The java.awt package provides classes for AWT api such as Text Field, Label, Text Area, Radio Button, Checkbox, Choice, List etc.

–>JAVA SWING:

Java Swing is a part of Java Foundation Classes (JFC) that is used to create window-based applications. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java.
Unlike AWT, Java Swing provides platform independent and lightweight components.
The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

Chapter 4

DESIGN

4.SYSTEM DESIGN: Systems design is the process or art of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development. There is some overlap and synergy with the disciplines of systems analysis, systems architecture and systems engineering.

Although typically used in software engineering it is a rich language that can be used to model an application structures, behavior and even business processes.

Use Case Diagram: A use case diagram is a graphic

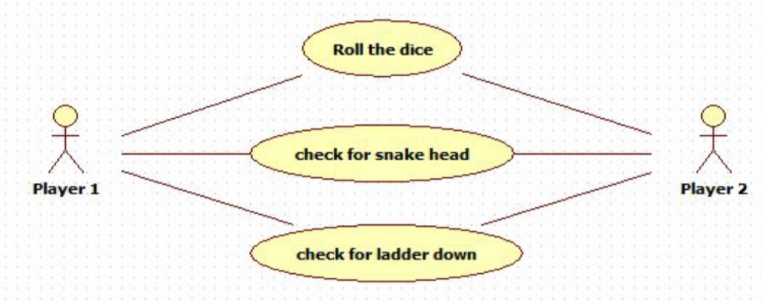

Figure 4.1: USE CASE DIAGRAM

depiction of the interactions among the elements of a system

Class Diagram: A class diagram is a type of static structure diagram that describes the structure of a system by showing the system's classes their attributes, operations and relationships among objects.

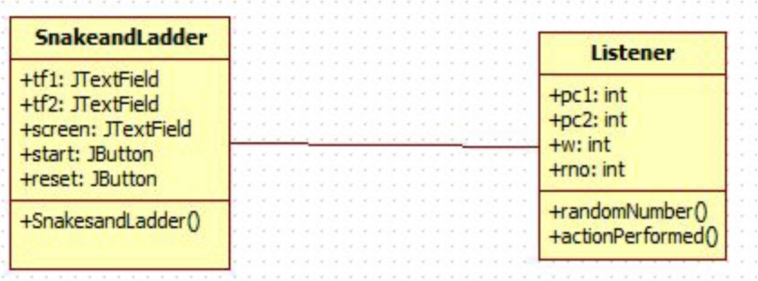

Figure 4.2: CLASS DIAGRAM

Sequence Diagram: A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence.

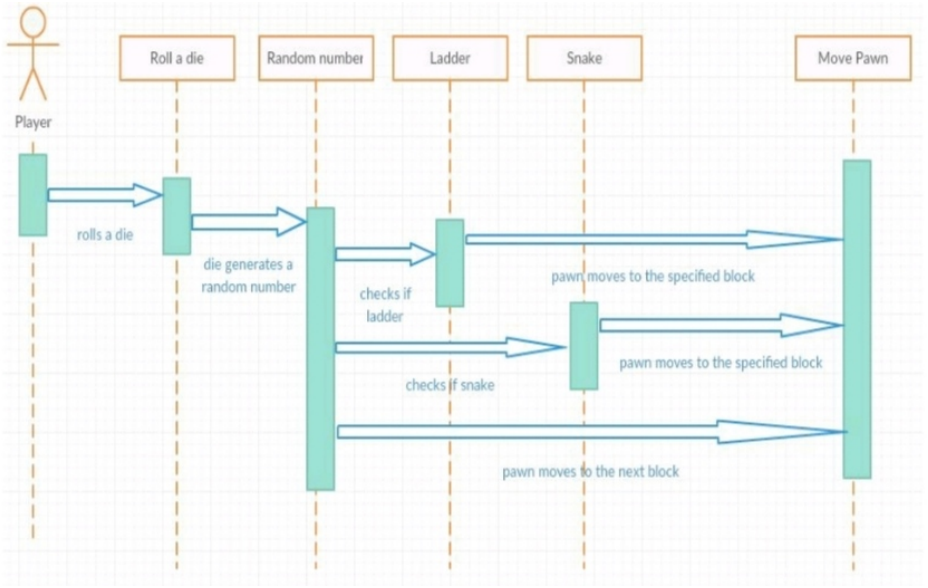

Figure 4.3: SEQUENCE DIAGRAM

OUTPUT SCREENSHOT:

Figure 4.4: OUTPUT SCREENSHOT 1

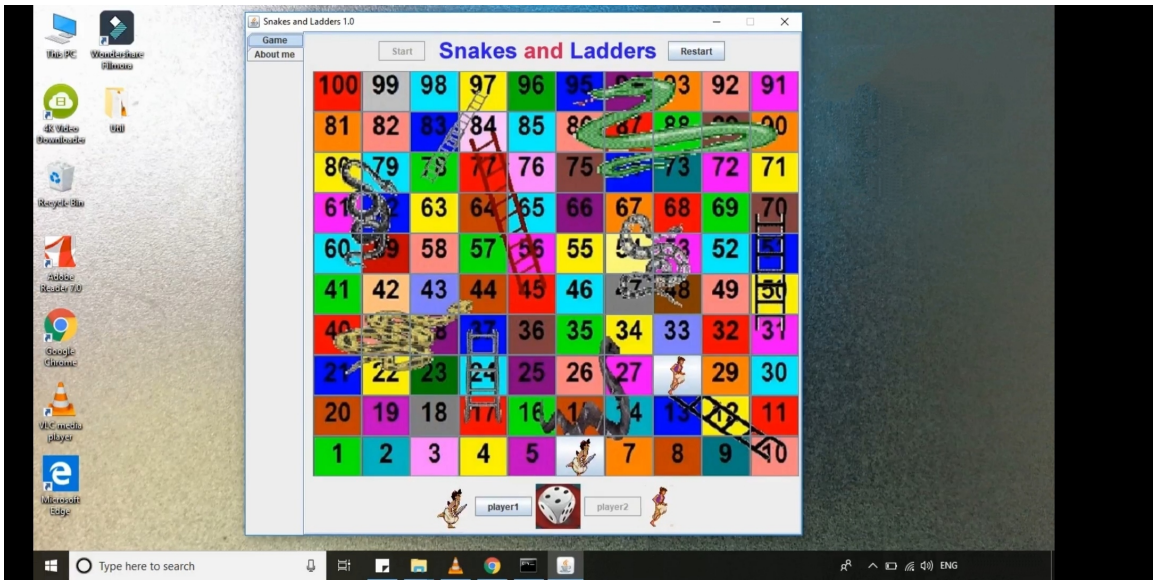

Figure 4.5: OUTPUT SCREENSHOT 2

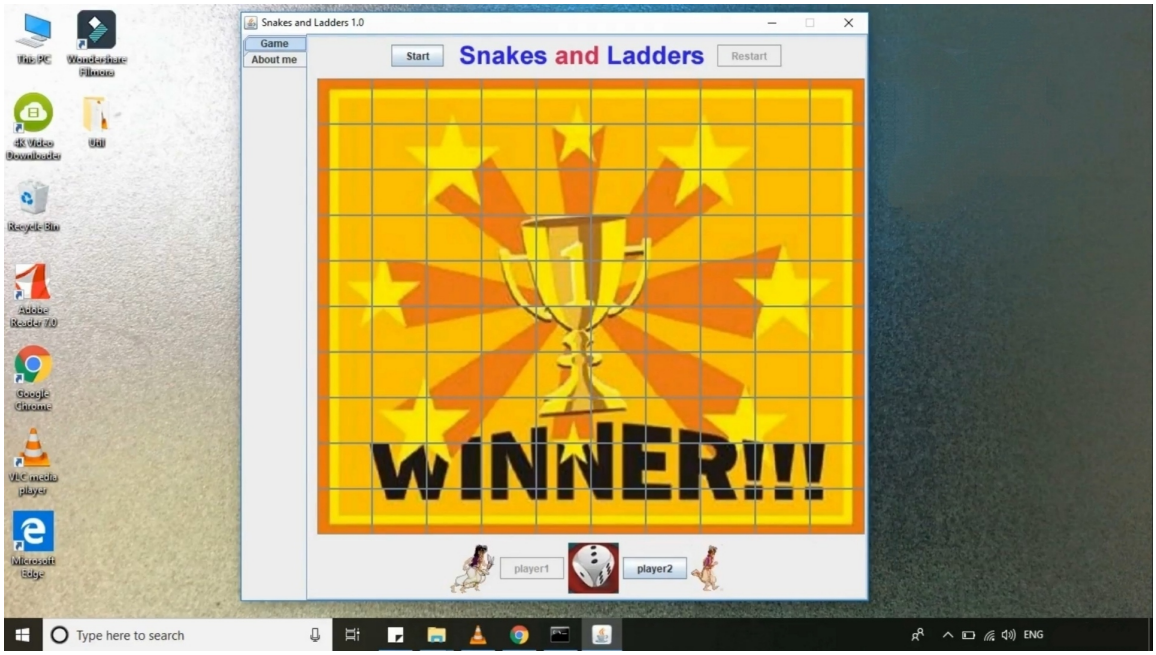

Figure 4.6: OUTPUT SCREENSHOT 3

Chapter 5

CONSLUSION

In this multi player snakes and ladders game, a minimum of two players and maximum of four players can play this game. In the beginning of the game, the player's tokens are placed in the start area. Every player will be given a chance and they have to go about it turn wise. The buttons are moved according to the number that is appeared on the rolled dice. Once the player's position is at the bottom of the ladder then he can climb it and reach to the top of the ladder. If the player encounters a snake he has to retrace his steps back by coming to the square where the tail of the snake is there. The player is won when the button of the player reaches the finish point first is declared as winners.

Appendices

SOURCE CODE

```
import javax.swing.*;
import javax.swing.event.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.net.*;
import java.io.*;

class Main implements ActionListener
{
 JFrame f;
 JTabbedPane tabPane;
 JPanel mainPanel,introPanel,gamePanel,playerPassPanel,
    ↪ diePanel;
 JPanel gameCenter,gameEast,gameWest,gameNorth,
    ↪ gameSouth;



 Icon p1,p2,header;
 Icon up,down;

 Icon icon[][]= new Icon[10][10];
 Icon winericon[][]= new Icon[10][10];

 Icon dieIcon;



 JButton introB[]= new JButton[5];
 JButton b[][]=new JButton[10][10];
 JButton start,restart;
 JButton JBplayer,JBcomputer;
```

```
JLabel die;



Random randomNo;


int imageFlag;
int i,j,num;
int prevIp1,prevJp1;

int path;
int p1value,p2value;

int player,computer;

int gameover;
int cimageFlag=0;
int cnoFlag=0;


String str;




 int n[][]={    {0, 1, 2, 3, 4, 5, 6, 7, 8, 9},
                {10,11,12,13,14,15,16,17,18,19},
                {20,21,22,23,24,25,26,27,28,29},
                {30,31,32,33,34,35,36,37,38,39},
                {40,41,42,43,44,45,46,47,48,49},
                {50,51,52,53,54,55,56,57,58,59},
```

```
                {60,61,62,63,64,65,66,67,68,69},
                {70,71,72,73,74,75,76,77,78,79},
                {80,81,82,83,84,85,86,87,88,89},
                {90,91,92,93,94,95,96,97,98,99},

        };

int game[][]={
               {100,99,98,97,96,95,94,93,92,91},
                {81,82,83,84,85,86,87,88,89,90},
                {80,79,78,77,76,75,74,73,72,71},
                {61,62,63,64,65,66,67,68,69,70},
                {60,59,58,57,56,55,54,53,52,51},
                {41,42,43,44,45,46,47,48,49,50},
                {40,39,38,37,36,35,34,33,32,31},
                {21,22,23,24,25,26,27,28,29,30},
                {20,19,18,17,16,15,14,13,12,11},
                { 1, 2, 3, 4, 5, 6, 7, 8, 9,10},
              };


int winer[][]={
                { 1, 2, 3, 4, 5, 6, 7, 8, 9,10},
                {11,12,13,14,15,16,17,18,19,20},
                {21,22,23,24,25,26,27,28,29,30},
                {31,32,33,34,35,36,37,38,39,40},
                {41,42,43,44,45,46,47,48,49,50},
                {51,52,53,54,55,56,57,58,59,60},
                {61,62,63,64,65,66,67,68,69,70},
                {71,72,73,74,75,76,77,78,79,80},
                {81,82,83,84,85,86,87,88,89,90},
                {91,92,93,94,95,96,97,98,99,100},
              };
```

```
Main()
{

f= new JFrame("Snakes and Ladders 1.0");
f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

mainPanel= new JPanel();
tabPane =new JTabbedPane(JTabbedPane.LEFT);


f.setLayout(new BorderLayout());

for(int i=0;i<introB.length;i++)
        {
         introB[i]= new JButton(Integer.toString(i));
        }

randomNo= new Random();

JBplayer= new JButton("player1");
JBcomputer= new JButton("player2");

JBplayer.setEnabled(false);
JBcomputer.setEnabled(false);


JBplayer.addActionListener(this);
JBcomputer.addActionListener(this);

die = new JLabel();

game();
```

```
introduction();


    f.add(tabPane,BorderLayout.CENTER);
 f.setResizable(false);
    f.setSize(750,700);
 //f.pack();
  f.setVisible(true);


}


void introduction()
{

 introPanel= new JPanel();

 JPanel main= new JPanel();
 JLabel l1= new JLabel();


 JLabel l2=new JLabel("<html><body><Font color='green'
    ↪ Size='6'>DONE BY: T.SHASHIPREETHAM REDDY<br>S.
    ↪ SAI GNANESHWAR REDDY.</Font><Font color='green'
    ↪ Size='4'><br></Font> <Font color='red' Size
    ↪ ='6'><br>**********thank you</Font>  <
    ↪ Font color='gray' Size='6'>**********</
    ↪ Font> </body></html>");


 Icon mypic;
```

```
 mypic=new ImageIcon("images/itsme.jpg");

 l1.setIcon(mypic);

 introPanel.setLayout(new BorderLayout());
 introPanel.add(l1,BorderLayout.CENTER);
 introPanel.add(l2,BorderLayout.SOUTH);



 tabPane.addTab("About me",introPanel);


}




void game()
{
 JLabel gameheader= new JLabel();
 JLabel logo = new JLabel();
 JLabel l1= new JLabel();
 JLabel l2= new JLabel();

 restart= new JButton("Restart");
 restart.setEnabled(false);
 restart.addActionListener(this);

 start= new JButton("Start");
 start.addActionListener(this);

 gamePanel= new JPanel();
```

```
gameCenter=new JPanel();
gameWest= new JPanel();
gameNorth=new JPanel();
gameSouth= new JPanel();

 gameCenter.setLayout(new GridLayout(10,10));
 gameEast= new JPanel(new GridLayout(2,1));

 gameWest.setLayout(new FlowLayout());
gameNorth=new JPanel(new FlowLayout());
gameSouth= new JPanel(new FlowLayout());

gamePanel.setLayout(new BorderLayout());

p1= new ImageIcon("images/p1.gif");
p2= new ImageIcon("images/p2.gif");
up= new ImageIcon("images/up.gif");
down= new ImageIcon("images/down.gif");

l1.setIcon(p1);
l2.setIcon(p2);

header=new ImageIcon("images/header.jpg");
gameheader.setIcon(header);

 for(int i=0;i<10;i++)
        for(int j=0;j<10;j++)
        {
                        b[i][j]= new JButton();
                //b[i][j].addActionListener(this);
                        path=game[i][j];
```

```
                        str=Integer.toString(path);
                        icon[i][j]=new ImageIcon("
                            ↪ images/"+str+".jpg");
                b[i][j].setIcon(icon[i][j]);

        }



for(int i=0;i<10;i++)
        for(int j=0;j<10;j++)
        {
                        path=winer[i][j];
                        str=Integer.toString(path);
                        winericon[i][j]=new ImageIcon("
                            ↪ images/winer/"+str+".jpg"
                            ↪ );
           // b[i][j].setIcon(winericon[i][j]);

        }



 str=Integer.toString(1);
 dieIcon= new ImageIcon("images/Game_dies/"+str+".jpg")
    ↪ ;
 die.setIcon(dieIcon);


 gameSouth.add(l1);
 gameSouth.add(JBplayer);
 gameSouth.add(die);
 gameSouth.add(JBcomputer);
```

```
 gameSouth.add(l2);

 gameNorth.add(start);
 gameNorth.add(gameheader);
 gameNorth.add(restart);

  for(int  i=0;i<10;i++)
        {
        for(int  j=0;j<10;j++)
        {
                gameCenter.add(b[i][j]);

              //System.out.print("["+i+"]["+j+"],");

        }
                //System.out.println();
        }



 gamePanel.add(gameCenter,BorderLayout.CENTER);
 gamePanel.add(gameWest,BorderLayout.WEST);
 gamePanel.add(gameNorth,BorderLayout.NORTH);
 gamePanel.add(gameSouth,BorderLayout.SOUTH);

  tabPane.addTab("Game",gamePanel);

}



void chance()
{
```

```
int n=randomNo.nextInt(3);
int i=0;

//System.out.println("Chacne is :"+n) ;
if(n==0)
        chance();

if(n==1)
        {
        player=1;
        computer=0;
        }
 else if(n==2)
        {
         player=0;
         computer=1;
        }




 if(n==1)
        {
        JBcomputer.setEnabled(false);
        JBplayer.setEnabled(true);

        }
else if(n==2)
        {
        JBcomputer.setEnabled(true);
        JBplayer.setEnabled(false);

        }
```

```
}




public void actionPerformed(ActionEvent e)
{

 int n=0;
try{
 if(e.getSource()==JBplayer)
        {

  //System.out.println("JB Player");
                    do{
                    n =playerPassNumber();
                    }while(n==0);

                    //System.out.println("UR playerPass
                       ↪ value is :"+n);
                    p1value=p1value+n;

                     if(p1value>=100)
                     {
                             p1value=100;
                     }
```

```
                playerimageTraval(p1value);
                    ↪

                JBcomputer.setEnabled(true);
                JBplayer.setEnabled(false);
                setBothImage();
        }
else  if(e.getSource()==JBcomputer)
        {
        //System.out.println("JB Computer");
                do{
                n =playerPassNumber();
                }while(n==0);

                //System.out.println("Computer Pass
                    ↪ Value is "+n);
                p2value=p2value+n;
                if(p2value>=100)
                {
                 p2value=100;
                }
                computerimageTraval(p2value);
                JBcomputer.setEnabled(false);
                JBplayer.setEnabled(true);

                setBothImage();
        }
else  if(e.getSource()==restart)
        {
         p1value=0;
         p2value=0;

        cnoFlag=0;
```

```
        cimageFlag=0;

         chance();
         rePrint();
        }
  else if(e.getSource()==start)
        {
                gameover=0;
                p1value=0;
                p2value=0;
                cnoFlag=0;
        cimageFlag=0;
                rePrint();
          start.setEnabled(false);
          restart.setEnabled(true);
          chance();
        }
        else
        {
        imageChange(e);
        }
}catch(Exception ee)
        {
                //System.out.println("Error in BUTTON
                    ↪ :"+ee);
        }
}



int playerPassNumber()
{
 int i= randomNo.nextInt(7);
```

```
            str=Integer.toString(i);
            dieIcon= new ImageIcon("images/Game_dies/"+str
                ↪ +".jpg");
            die.setIcon(dieIcon);

        return i;

}




void winerdraw()
{
int i=0;
                    start.setEnabled(true);
                    restart.setEnabled(false);
                    JBplayer.setEnabled(false);
                    JBcomputer.setEnabled(true);
                    JBcomputer.setEnabled(false);

for(int k=0;k<10;k++)
            for(int l=0;l<10;l++)
            {
                        gameover=1;
             b[k][l].setIcon(winericon[k][l]);
            }


}
```

```
void rePrint()
{
try{

for( i=0;i<10;i++)
        {
        for(j=0;j<10;j++)
          {
                    // b[i][j].setIcon(null);
                        b[i][j].setIcon(icon[i][j]);
          }
        }

   }catch(Exception e)
        {
        //System.out.println("error in repaint"+e);
        }

}




void playerimageTraval(int n)
{
int i=j=0;
int imageFlag=0;
int noFlag=0;

if(gameover==0)
{
try{
```

```
        rePrint();
          for(i=0;i<10;i++)
                {
                  for(j=0;j<10;j++)
                        {
                          if(n==game[i][j])
                              {
                 noFlag=1;
                                    break;
                              }
                        }
                 if(noFlag==1)
                        break;
                }

if(noFlag==1)
                {
                if(imageFlag==0)
                {
                 Thread.sleep(200);
                 b[i][j].setIcon(p1);
                 imageFlag=1;
                 prevIp1=i;
                 prevJp1=j;
                }
      else
                {
                path=game[prevIp1][prevJp1];
                str=Integer.toString(path);
                Thread.sleep(200);
                icon[prevIp1][prevJp1]=new ImageIcon("
                    ↪ images/"+str+".jpg");
                Thread.sleep(200);
```

```
            b[prevIp1][prevJp1].setIcon(icon[
               ↪ prevIp1][prevJp1]);
            b[i][j].setIcon(p1);
            prevIp1=i;
            prevJp1=j;
            }

        /*****************checks for staires
           ↪ **********************/
            if(n==10)
                    {
                    n=28;
                    playerimageTraval(n);
                    p1value=n;
                    }
            else if(n==17)
                    {
                    n=37;
                    playerimageTraval(n);
                    p1value=n;
                    }
            else if(n==31)
                    {
                    n=70;
                    playerimageTraval(n);
                    p1value=n;
                    }
            else if(n==45)
                    {
                    n=84;
                    playerimageTraval(n);
                    p1value=n;
                    }
```

```
else if(n==78)
        {
        n=97;
        playerimageTraval(n);
        p1value=n;
        }


/***************Checks for Snakes
   ↪ *************************/
if(n==95)
        {
         n=73;
         playerimageTraval(n);
         p1value=n;
             ↪
        }
else if(n==79)
        {
         n=59;
         playerimageTraval(n);
         p1value=n;
             ↪
        }
else if(n==68)
        {
         n=48;
         playerimageTraval(n);
         p1value=n;
             ↪
        }
else if(n==44)
        {
```

```
                        n=21;
                        playerimageTraval(n);
                        p1value=n;
                            ↪
                    }
                else if(n==34)
                    {
                        n=16;
                        playerimageTraval(n);
                        p1value=n;
                            ↪
                    }

                }

        if(n>=100)
        {
                //System.out.println("Congr u r win");
                winerdraw();

        }
}
catch(Exception e)
{

}
}

}

void setBothImage()
{
int i=0;
```

```
int j=0;
int noFlag=0;

if(gameover==0)
{
for(i=0;i<10;i++)
                    {
                      for(j=0;j<10;j++)
                                {
                                  if(p1value==game[i][j])
                                      {
                                            noFlag=1;
                                            break;
                                      }
                                }
                       if(noFlag==1)
                                break;
                    }

if(noFlag==1)
{
b[i][j].setIcon(p1);
}

noFlag=0;

for(i=0;i<10;i++)
                    {
                      for(j=0;j<10;j++)
                                {
                                  if(p2value==game[i][j])
                                      {
                                            noFlag=1;
```

```
                                        break;
                                    }
                                }
                            if(noFlag==1)
                                    break;
                        }

if(noFlag==1)
{
b[i][j].setIcon(p2);
}
}
}


void computerimageTraval(int n)
{
int i=j=0;
int imageFlag=0;
int noFlag=0;

try{
          rePrint();
            for(i=0;i<10;i++)
        {
                    for(j=0;j<10;j++)
                            {
                              if(n==game[i][j])
                                {
                    noFlag=1;
                                        break;
                                }
                            }
```

```
            if(noFlag==1)
                    break;
            }

    if(noFlag==1)
            {
            if(imageFlag==0)
            {
             Thread.sleep(200);
             b[i][j].setIcon(p2);
             imageFlag=1;
             prevIp1=i;
             prevJp1=j;
            }
else
            {
            path=game[prevIp1][prevJp1];
            str=Integer.toString(path);
            Thread.sleep(200);
            icon[prevIp1][prevJp1]=new ImageIcon("
               ↪ images/"+str+".jpg");
            Thread.sleep(200);
            b[prevIp1][prevJp1].setIcon(icon[
               ↪ prevIp1][prevJp1]);
            b[i][j].setIcon(p1);
            prevIp1=i;
            prevJp1=j;
            }


        /*****************checks for staires
           ↪ **********************/
```

```
if(n==10)
        {
        n=28;
        computerimageTraval(n);
        p2value=n;
        }
else if(n==17)
        {
        n=37;
        computerimageTraval(n);
        p2value=n;
        }
else if(n==31)
        {
        n=70;
        computerimageTraval(n);
        p2value=n;
        }
else if(n==45)
        {
        n=84;
        computerimageTraval(n);
        p2value=n;
        }
else if(n==78)
        {
        n=97;
        computerimageTraval(n);
        p2value=n;
        }
```

```
/*****************Checks for Snakes
    ↪ *************************/
if(n==95)
        {
         n=73;
         computerimageTraval(n);
         p2value=n;
             ↪
        }
else if(n==79)
        {
         n=59;
         computerimageTraval(n);
         p2value=n;
             ↪
        }
else if(n==68)
        {
         n=48;
         computerimageTraval(n);
         p2value=n;
             ↪
        }
else if(n==44)
        {
         n=21;
         computerimageTraval(n);
         p2value=n;
             ↪
        }
else if(n==34)
        {
         n=16;
```

```
                                computerimageTraval(n);
                                p2value=n;
                                    ↪
                               }

                        }


                if(n>=100)
                {
                        //System.out.println("Congr u r win");
                        gameover=1;
                        winerdraw();
                }



}
catch(Exception e)
{

}


}

void imageChange(ActionEvent e)
{
int i=j=0;
int flag=0;

try{
 for( i=0;i<10;i++)
```

```
{
    for( j=0;j<10;j++)
    {
     if(e.getSource()==b[i][j])
            {
                    //System.out.println("Button
                        ↪ Found");
                    //System.out.println("b["+i
                        ↪ +"]["+j+"]");
                    flag=1;
                    break;
            }
    //System.out.println(j);
     }

     if(flag==1)
            break;
 }


    if(imageFlag==0)
    {
             Thread.sleep(200);
             b[i][j].setIcon(p1);
             imageFlag=1;
            prevIp1=i;
            prevJp1=j;
            }
else
    {
            path=game[prevIp1][prevJp1];
            str=Integer.toString(path);
            Thread.sleep(200);
```

```
                    icon[prevIp1][prevJp1]=new ImageIcon("
                        ↪ images/"+str+".jpg");
                            Thread.sleep(200);
                    b[prevIp1][prevJp1].setIcon(icon[
                        ↪ prevIp1][prevJp1]);
                    b[i][j].setIcon(p1);
                    prevIp1=i;
                    prevJp1=j;
            }
        }
catch(Exception ee)
            {
                    ee.printStackTrace();
            }
}




public static void main(String args[])
{
 Main m= new Main();
}

}
```