/\*\*

Program: Assignment 2: Application – Ball Maze

Filename: CBallMaze.java

@author: © Bhuwan Khatiwada.

UN ID: 18406498

Course: BSc. (Hons) Computing Year 1

Module: CSY1020 Problem Solving & Programming

Date: 20th July 2018

\*/

All the code written to acquire final solution for this assignment can be found inside class named as CBallMaze. The code written under class CBallMaze can be found below.

**Class CBallMaze**

//importing all necessary classes and function needed for this class.

import javax.sound.sampled.AudioInputStream;

import javax.sound.sampled.AudioSystem;

import javax.sound.sampled.Clip;

import javax.sound.sampled.LineUnavailableException;

import javax.sound.sampled.UnsupportedAudioFileException;

import javax.swing.\*; //calling method to create a JFame.

import java.awt.event.\*;

import java.io.File;

import java.io.IOException;

import java.net.URL;

import java.util.Random;

import javax.swing.GroupLayout.Alignment;

import javax.swing.border.EmptyBorder;

import javax.swing.event.ChangeEvent;

import javax.swing.event.ChangeListener;

import java.awt.\*;

import java.awt.event.\*;

// class named CBallMaze is created which is extended as JFrame.

public class CBallMaze extends JFrame implements ActionListener, KeyListener {

//creating JMenuBar in the Swing

//creating all menu for application.

private static JMenuBar menubar;

private static JMenu scenario;

private static JMenu edit;

private static JMenu control;

private static JMenu help;

//creating all menu-items for application.

private static JMenuItem newf,open,openrecent,save, saveall,exit; //menu items for menu scenario

private static JMenuItem newcl,opencl,copy,cut, undo,paste; //menu items for menu edit

private static JMenuItem run1,act1,reset1; //menu items for menu control.

private static JMenuItem about,help1; //menu items for menu help.

//creating panels in JFrame

private static JPanel PMaze,PButtom,PRight;

//creating JLabel for panel1

private static JLabel jLball,jLgoal,jLscore,jLcount; //jlabel for adding ball and goal to the scenario

private static JLabel jLsands[][]= new JLabel[(16)][(13)]; //2D array created for adding maze

//asssigning GridBagConstraint for panel1, PanelMaze

GridBagConstraints c = new GridBagConstraints();//getting GridBagLayout's GridBagConstraints

GridBagConstraints gc = new GridBagConstraints();//getting GridBagLayout's GridBagConstraints for moving apple icon.

//creating jbutton for panel2

private static JButton jBact,jBrun,jBreset,jBpause;

//creating JLabel for panel2

private static JLabel jLspeed;

// Timer assigned for panel2 (PRight)

private Timer digitalclock; //Timer for digital clock.

//creating javax timer to ruler/slider

private javax.swing.Timer slider;

//creating timer for actions in panel1(PMaze)

Timer run; //timer for run function.

Timer change; //timer for option 3 method.

// creating a int ticks to make our timer works.

private int nticks = 0; //initializing global varibale for digital clock.

//creating JButton for panel3 (PRight)

private static JButton jBoption1,jBoption2,jBoption3,jBoption4,jB1,jB2,jB3,jB4,jB5,jB6,jB7,jB8,jB9,jBcompass;

//creating JLabel for panel3(PRight)

private static JLabel jLoption,jLsquare,jLdirection,jLtimer,jLsep,jLsep1,jLemt1,jLemt2,jLempt3;

//creating JTextFiled for panel3(PRight)

private static JTextField jTone,jTtwo,jTthree,jTclock,jTclock1,jTclock2;

// creating slider for panel2(PButtom)

private static JSlider Jruler;

// initializing a variable for calculating number of square travelled.

int nsq = 2;

//importing images in jframe inside jpanel1

ImageIcon icon\_sandroad = new ImageIcon("pictures\\sand.jpg"); //image of sand road.

ImageIcon icon\_goldball = new ImageIcon("pictures\\gold-ball.png"); //image of golden ball.

ImageIcon icon\_sandstone = new ImageIcon("pictures\\sandstone.jpg"); //image of goal.

ImageIcon icon\_whitespace = new ImageIcon("pictures\\white32x32.jpg"); //image of whitespace.

ImageIcon icon\_apple = new ImageIcon("pictures\\apple.png"); //image of fruit apple.

//importing compass images for panel3 or panel to the right.

ImageIcon icon\_pictures = new ImageIcon("pictures\\north.jpg"); //image of north direction.

ImageIcon icon\_pictures1 = new ImageIcon("pictures\\south.jpg"); //image of south direction.

ImageIcon icon\_pictures2 = new ImageIcon("pictures\\east.jpg"); //image of east direction.

ImageIcon icon\_pictures3 = new ImageIcon("pictures\\west.jpg"); //image of west direction.

//starting method to work on JFrame

public static void main(String[] args) {

//managing JFrame size,title and other feature

CBallMaze frame = new CBallMaze ();//creating new java frame

ImageIcon greenfoot = new ImageIcon("pictures\\greenfoot.png"); //image icon for JFrame.

frame.setSize(775,650); //setting default size of frame

frame.setIconImage(greenfoot.getImage()); //setting image icon for jFrame.

frame.setTitle("CBallMaze- Ball Maze Application");//title of Jframe

frame.createGUI(); //creating method createGUI();

//creating menubar in the JFrame

menubar = new JMenuBar();//creating menubar

scenario = new JMenu("Scenario");//creating menu items

edit = new JMenu("Edit");//creating menu items

control = new JMenu("Control");//creating menu items

help = new JMenu("Help");//creating menu items

//adding menu items to the scenario

menubar.add(scenario);//adding first items

menubar.add(edit);//adding second items

menubar.add(control);//adding third items

menubar.add(help);//adding fourth items

// adding sub menu items to scenario menu items

newf = scenario.add("New"); //adding sub menu to menu.

open = scenario.add("Open");//adding sub menu to menu.

openrecent = scenario.add("Open Recent");//adding sub menu to menu.

save = scenario.add("Save");//adding sub menu to menu.

saveall = scenario.add("Save All");//adding sub menu to menu.

exit = scenario.add("Exit");//adding sub menu to menu.

//adding action listner to exit menu item.

exit.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

System.exit(0); // exiting Frame when exit menu item is pressed.

}

});

// adding sub menu items to edit menu items

newcl = edit.add("New Class");//adding sub menu to menu.

opencl = edit.add("Import Class");//adding sub menu to menu.

copy = edit.add("Copy");//adding sub menu to menu.

cut = edit.add("Cut");//adding sub menu to menu.

undo = edit.add("Undo");//adding sub menu to menu.

paste = edit.add("Paste");//adding sub menu to menu.

//adding sub menu to control menu items

run1 = control.add("Run");//adding sub menu to menu.

act1 = control.add("Act");//adding sub menu to menu.

reset1 = control.add("Reset");//adding sub menu to menu.

//adding sub menu to help menu items

about = help.add("About");//adding sub menu to menu.

help1 = help.add("Help");//adding sub menu to menu.

//adding action to about menu items

about.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

JOptionPane.showMessageDialog(null,"Program: Assignment 2: Application – Ball Maze \n Filename: CBallMaze.java \n @author: © Bhuwan Khatiwada. \n UN ID : 18406498 \n Course: BSc. (Hons) Computing Year 1 \n Module: CSY1020 Problem Solving & Programming\n Date: 20th July 2018" );

}

});

//adding menubar to JFrame

frame.setJMenuBar(menubar);//adding whole menubar in the Jframe

//performing other necessary functions for JFrame.

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);//closing jframe when press exit

frame.setLocationRelativeTo(null);//displaying jframe at the center of the screen.

frame.setVisible(true);//making jframe visible

frame.setResizable(false);//making window frame non resizable

}

// creating private void createGUI to add different panels

private void createGUI() {

Container back = getContentPane(); //making a container to get all ContentPane

//adding panel for movement of the ball

PMaze = new JPanel(); //first panel to create a maze

PMaze.setLayout(new GridBagLayout()); //GridBagLayout is created

PMaze.setBackground(Color.WHITE); //setting Panel Background White

PMaze.setBorder(BorderFactory.createLineBorder(Color.black)); //Creating black border around the panel.

//adding goldball in the panel1

c.gridx = 15;

c.gridy = 0;

jLball = new JLabel(icon\_goldball); //making an JLabel for ball.

PMaze.add(jLball,c); //adding ball.

// adding Goal in the panel1

c.gridx =0;

c.gridy=12;

jLgoal = new JLabel(icon\_sandstone);// makingn JLabel for goal.

PMaze.add(jLgoal,c);//adding goal.

//creating jlabel inside jpanel1

//adding all row of sandroad in the panel.

for(c.gridy =0;c.gridy<13;c.gridy+=3) {

for (c.gridx = 0; c.gridx <16; c.gridx++) {

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ; //making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c); //adding all JLabels.

}

}

//adding sandroad vertivally below 1st row

for (c.gridy=1;c.gridy<3;c.gridy++) {

c.gridx=1;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);

}

for (c.gridy=1;c.gridy<3;c.gridy++) {

c.gridx=5;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);

}

for (c.gridy=1;c.gridy<3;c.gridy++) {

c.gridx=9;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

//adding sandroad vertivally in x-axis = 2 for 4th row

for (c.gridy=4;c.gridy<6;c.gridy++) {

c.gridx=2;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

//adding sandroad vertivally in x-axis = 10 for 4th row

for (c.gridy=4;c.gridy<6;c.gridy++) {

c.gridx=6;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

//adding sandroad vertivally in x-axis = 10 for 4th row

for (c.gridy=4;c.gridy<6;c.gridy++) {

c.gridx=11;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

//adding sandroad vertivally in x-axis = 11 for 7th row

for (c.gridy=7;c.gridy<9;c.gridy++) {

c.gridx=1;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

//adding sandroad vertivally for 7th row

for (c.gridy=7;c.gridy<9;c.gridy++) {

c.gridx=5;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

//adding sandroad vertivally for 7th row

for (c.gridy=7;c.gridy<9;c.gridy++) {

c.gridx=12;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

//adding sandroad vertivally for 7th row

for (c.gridy=10;c.gridy<12;c.gridy++) {

c.gridx=3;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

//adding sandroad vertivally for 9th row

for (c.gridy=10;c.gridy<12;c.gridy++) {

c.gridx=6;

jLsands[c.gridx][c.gridy] = new JLabel(icon\_sandroad) ;//making arrays of JLabel.

PMaze.add( jLsands[c.gridx][c.gridy],c);//making arrays of JLabel.

}

// adding jLscore at the end.

c.gridx = 8; c.gridy = 7;

jLscore = new JLabel("Score"); //adding jLscore

PMaze.add(jLscore,c); //adding Jlabel to panel 1.

jLscore.setVisible(false);

//adding score counter

c.gridx = 9; c.gridy = 7;

jLcount = new JLabel("0"); //adding jLscore

PMaze.add(jLcount,c); //adding Jlabel to panel 1.

jLcount.setVisible(false);

//making grid-x and grid-y back to 15 and 0 respectively.

c.gridx = 15;

c.gridy = 0;

//panel1 action completed

//adding panel2 at button of JFrame

PButtom = new JPanel(new FlowLayout()); //adding panel2 with flow layout

PButtom.setBorder(BorderFactory.createLineBorder(Color.black)); //setting border around Jpanel2

//adding panel at right of the JFrame

PRight = new JPanel(); //adding panel3 in JFrame

PRight.setPreferredSize(new Dimension(160,748)); //setting size of panel3 in JFrame

PRight.setBorder(BorderFactory.createLineBorder(Color.black)); //setting border around JPanel3

//adding all three panels to the JFrame

back.add(PMaze,BorderLayout.CENTER); //adding panel1 in JFrame

back.add(PButtom,BorderLayout.PAGE\_END); //adding panel2 in JFrame

back.add(PRight,BorderLayout.EAST); //adding panel3 in Jframe

//working in panel2

//adding image to JButton in panel 2

ImageIcon icon\_actimg = new ImageIcon("pictures\\step.png"); //importing picture for act button

ImageIcon icon\_runimg = new ImageIcon("pictures\\run.png"); //importing picture for run button.

ImageIcon icon\_resetimg = new ImageIcon("pictures\\reset.png"); //importing picture for reset button.

ImageIcon icon\_pause = new ImageIcon("pictures\\pause.png"); //adding a pause icon to button.

// button for PButtom along with the icon.

jBact = new JButton("ACT",icon\_actimg); //making button for act

jBrun = new JButton("RUN",icon\_runimg); //making button for run

jBreset = new JButton("RESET",icon\_resetimg); //making button for reset.

jBpause = new JButton("Pause",icon\_pause); //making a pause button.

//slider for PButtom

Jruler = new JSlider(JSlider.HORIZONTAL,0,50,25); //silder with holizontal line starting form 0 and ends at 50.

Jruler.setMajorTickSpacing(10); //tick spacing set to 10.

// Jruler.setPaintLabels(true); //labelling ticks in slider

Jruler.setPaintTicks(true); //making ticks in slider.

Jruler.setSnapToTicks(true);//directly jumps to next ticks.

Jruler.setInverted(true); //making the value of slider inverted.

//adding button to PButtom

PButtom.add(jBact); //adding button act.

PButtom.add(jBrun); //adding button run.

PButtom.add(jBpause); //adding a pause button.

PButtom.add(jBreset); //adding button reset.

jBpause.setVisible(false); //setting pause button as false.

//creating space between JButton to JSlider

PButtom.add(Box.createRigidArea(new Dimension(200, 0)));//setting space between JButton and JSlider

// JLabel for panel2

jLspeed = new JLabel ("Speed :"); //JLabel assigned.

//adding JLabel in panel 2

PButtom.add(jLspeed); //adding JLabel speed.

jLspeed.setFont(new Font("Times New Roman", Font.BOLD, 16) );//making content of JLabel bold.

//adding slider to panel 2

PButtom.add(Jruler); //adding a ruler.

//actions on JPanel2 completed.

//working in JPanel3

//image for pause button in panel3

//Button for panel3

jBoption1 = new JButton("Option1"); //addding a button.

jBoption2 = new JButton("Option2");//addding a button.

jBoption3 = new JButton("Option3");//addding a button.

jBoption4 = new JButton("Exit");//addding a button.

jB1 = new JButton();//addding a button.

jB2 = new JButton("^");//addding a button.

jB3 = new JButton();//addding a button.

jB4 = new JButton("<");//addding a button.

jB5 = new JButton();//addding a button.

jB6 = new JButton(">");//addding a button.

jB7 = new JButton();//addding a button.

jB8 = new JButton("v");//addding a button.

jB9 = new JButton();//addding a button.

// JTextField for JPanel3

jTone = new JTextField(6); //textfield for size 10.

jTone.setText("1"); //text set to 1.

jTone.setHorizontalAlignment(JTextField.CENTER);

jTtwo = new JTextField(6); //textfield for size 10.

jTtwo.setText("1"); //text set to 1.

jTtwo.setHorizontalAlignment(JTextField.CENTER); //aligning text to center.

jTthree = new JTextField(6); //textfield for size 10.

jTthree.setText("W"); //text set to W.

jTthree.setHorizontalAlignment(JTextField.CENTER); //aligning text to center.

jTclock = new JTextField(3); //textfield for size 3.

jTclock.setText("00"); //text set to 00.

jTclock.setHorizontalAlignment(JTextField.CENTER);//aligning text to center.

jTclock1 = new JTextField(3); //textfield for size 3.

jTclock1.setText("00");//text set to 00.

jTclock1.setHorizontalAlignment(JTextField.CENTER);//aligning text to center.

jTclock2 = new JTextField(3); //textfield for size 3.

jTclock2.setText("00");//text set to 00.

jTclock2.setHorizontalAlignment(JTextField.CENTER);//aligning text to center.

// Jlabel for the panel3

jLoption = new JLabel("Option"); //adding a label.

jLsquare = new JLabel("Square");//adding a label.

jLdirection = new JLabel("Direction");//adding a label.

jLtimer = new JLabel("DIGITAL TIMER");//adding a label.

jLsep = new JLabel (":");//adding a label.

jLsep1 = new JLabel (":");//adding a label.

jLemt1 = new JLabel();//adding a label.

jLemt2 = new JLabel();//adding a label.

//adding JLabel AND JTextField to JLabel3

PRight.add(jLoption); //adding Jlabel option

jLoption.setFont(new Font("Times New Roman", Font.BOLD, 16) ); // making font bold and of size 16.

PRight.add(Box.createRigidArea(new Dimension(8, 0))); //creating space of 8 unit.

PRight.add(jTone);//adding text field.

PRight.add(jLsquare);//adding text field.

jLsquare.setFont(new Font("Times New Roman", Font.BOLD, 16));// making font bold and of size 16.

PRight.add(Box.createRigidArea(new Dimension(9, 0)));//creating space of 8 unit.

PRight.add(jTtwo);//adding text field.

PRight.add(jLdirection); //adding text field.

jLdirection.setFont(new Font("Times New Roman", Font.BOLD, 16) );// making font bold and of size 16.

PRight.add(jTthree);//adding text field.

//adding digital clock in panel3

EmptyBorder empty1 = new EmptyBorder(10, 30, 1, 30); //creating an empty border.

PRight.add(jLtimer); //adding timer label.

jLtimer.setBorder(empty1); //setting an empty border.

jLtimer.setFont(new Font("Times New Roman", Font.BOLD, 18)); //settting font of timer label.

//making 3 TextFiled for digital clock

PRight.add(jTclock); //adding textfield for clock.

jTclock.setBackground(Color.BLACK); //background colour to black.

jTclock.setForeground(Color.WHITE); //background colour to white.

PRight.add(jLsep);//adding textfield for clock.

PRight.add(jTclock1);//adding textfield for clock.

jTclock1.setBackground(Color.BLACK); //background colour to black.

jTclock1.setForeground(Color.WHITE); //background colour to white.

PRight.add(jLsep1);//adding textfield for clock.

PRight.add(jTclock2);//adding textfield for clock.

jTclock2.setBackground(Color.BLACK); //background colour to black.

jTclock2.setForeground(Color.WHITE); //background colour to white.

PRight.add(Box.createRigidArea(new Dimension(0, 20))); //creating rigid area to make spacing.

//adding empty jlabel to panel 3

EmptyBorder empty2 = new EmptyBorder(0,20,25,100); //creating empty border empty2

jLemt1.setBorder(empty2); //setting empty border to JLabel emt1

PRight.add(jLemt1); //adding empty JLabeel to panel3

//adding button to panel3

PRight.add(jB1); //adding a button.

jB1.setPreferredSize(new Dimension(40, 40)); //defining size for button.

jB1.setBorder(BorderFactory.createLineBorder(Color.BLACK, 1)); //adding border to button.

jB1.setBackground(Color.WHITE);//background set to white

jB1.setEnabled(false);//disabling button.

PRight.add(jB2);//adding a button.

jB2.setPreferredSize(new Dimension(40, 40));//defining size for button.

jB2.setFont(new Font("Times New Roman", Font.PLAIN, 16) ); //defining font size and type

PRight.add(jB3);//adding a button.

jB3.setPreferredSize(new Dimension(40, 40));//defining size for button.

jB3.setBorder(BorderFactory.createLineBorder(Color.BLACK,1));//adding border to button.

jB3.setBackground(Color.WHITE);//background set to white

jB3.setEnabled(false);//disabling button.

PRight.add(jB4);//adding a button.

jB4.setPreferredSize(new Dimension(40, 40));//defining size for button.

jB4.setFont(new Font("Times New Roman", Font.PLAIN, 12) ); //defining font size and type

PRight.add(jB5);//adding a button.

jB5.setPreferredSize(new Dimension(40, 40));//defining size for button.

jB5.setBackground(Color.WHITE);//background set to white

jB5.setEnabled(false);//disabling button.

jB5.setBorder(BorderFactory.createLineBorder(Color.BLACK, 1));//adding border to button.

PRight.add(jB6);//adding a button.

jB6.setPreferredSize(new Dimension(40, 40));//defining size for button.

jB6.setFont(new Font("Times New Roman", Font.PLAIN, 12) ); //defining font size and type

PRight.add(jB7);//adding a button.

jB7.setPreferredSize(new Dimension(40, 40));//defining size for button.

jB7.setBorder(BorderFactory.createLineBorder(Color.BLACK, 1));//adding border to button.

jB7.setBackground(Color.WHITE);//background set to white

jB7.setEnabled(false);//disabling button.

PRight.add(jB8);//adding a button.

jB8.setPreferredSize(new Dimension(40, 40));//defining size for button.

jB8.setFont(new Font("Times New Roman", Font.PLAIN, 13) ); //defining font size and type

PRight.add(jB9);//adding a button.

jB9.setPreferredSize(new Dimension(40, 40));//defining size for button.

jB9.setBorder(BorderFactory.createLineBorder(Color.BLACK,1));//adding border to button.

jB9.setBackground(Color.WHITE); //background set to white

jB9.setEnabled(false); //disabling button.

//adding empty border in panel3 under JLabel emt2

jLemt2.setBorder(empty2); //setting an empty border.

PRight.add(jLemt2); //adding an empty border.

//adding different button to Jpanel3

PRight.add(jBoption1); //adding a button.

jBoption1.setPreferredSize(new Dimension(60, 30)); //defining dimension to the option buttons.

jBoption1.setFont(new Font("Times New Roman", Font.PLAIN, 8) ); //defining font size and type

PRight.add(jBoption2);//adding a button.

jBoption2.setPreferredSize(new Dimension(60, 30));//defining dimension to the option buttons.

jBoption2.setFont(new Font("Times New Roman", Font.PLAIN, 8) );//defining font size and type

PRight.add(jBoption3);//adding a button.

jBoption3.setFont(new Font("Times New Roman", Font.PLAIN, 8) );//defining font size and type

jBoption3.setPreferredSize(new Dimension(62, 26));//defining dimension to the option buttons.

PRight.add(jBoption4);//adding a button.

jLempt3 =new JLabel(); //new JLabel for empty border.

EmptyBorder empty3 = new EmptyBorder(0,4,10,100); //empty border initialized.

jLempt3.setBorder(empty3);//empty border created.

PRight.add(jLempt3); //empty border assigned.

//adding button for compass in JLabel

jBcompass = new JButton(icon\_pictures); //making new JButton for adding compass.

PRight.add(jBcompass); //adding compass button.

jBcompass.setPreferredSize(new Dimension(80,80));

jBcompass.setBackground(Color.LIGHT\_GRAY); //colour of button defined.

//adding actionlistener to arrow keys

jB2.addActionListener(this);

jB4.addActionListener(this);

jB6.addActionListener(this);

jB8.addActionListener(this);

jBoption4.addActionListener(this);

// adding action listener to option1 key

jBoption1.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

jTone.setText("1"); //text set to 1.

createGUI(); //calling function createGUI();

change.stop(); //stopping change timer.

}

});

//adding actionlistener to pause key

jBoption2.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

int ncount = 1; //initializing count variable

jTone.setText("2"); //text set to 2.

Random number = new Random(); // random number called.

for(int i = 0 ;i<13;i+=3) {

c.gridy=i; //assinging gridy value from the loop.

c.gridx=number.nextInt(13); //making an random number for gridx.

jLsands[c.gridx][c.gridy].setIcon(icon\_apple); //adding apple icon.

}

//defining position of gridx and gridy.

c.gridx=15;

c.gridy=0;

createGUI();//calling function createGUI();

change.stop();//stopping change timer.

}

});

//adding actionListener to btnoption3.

jBoption3.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

jTone.setText("3"); //text set to 2.

makeChange(); //calling method on button option3.

digitalclock(jTclock1, jTclock2); //calling method digital clock.

c.gridx=15;c.gridy=0;

}

});

//adding action listener to run button.

jBrun.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

runBall(); // calling function runBall.

jBpause.setVisible(true); //enabling pause button

jBrun.setVisible(false); //disabling run button.

jBact.setEnabled(false); //disabling button "act".

digitalclock(jTclock1, jTclock2); //calling a method.

}

});

//adding action listen to pause button.

jBpause.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

run.stop();

jBpause.setVisible(false); //disabling pause button.

jBrun.setVisible(true); //enabling run button.

jBact.setEnabled(true); //enabling act button.

}

});

//resetting the maze when reset button is pressed.

jBreset.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

createGUI(); //calling method createGUI();

run.stop();//stopping change run.

change.stop();//stopping change timer.

}

});

// making all operation to run under act function.

jBact.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

if((c.gridx==9 && c.gridy<3) || (c.gridx==6 && c.gridy<6) ||(c.gridx==5 && c.gridy<9)||(c.gridx==3 && c.gridy<12)) {

jLsands[c.gridx][c.gridy+1].setIcon(icon\_goldball); //making our animation of ball work.

jLsands[c.gridx][c.gridy].setIcon(icon\_sandroad);//making our animation of ball work.

c.gridy++; //increasing gridy by 1.

jTthree.setText("S"); //text set to S

jBcompass.setIcon(icon\_pictures1); //setting new icon.

jTtwo.setText(String.valueOf(nsq)); //setting new value.

++nsq; //increasing value of nsq.

}

else {

jLball.setIcon(icon\_sandroad);//making our animation of ball work.

jLsands[c.gridx-1][c.gridy].setIcon(icon\_goldball);//making our animation of ball work.

jLsands[c.gridx][c.gridy].setIcon(icon\_sandroad);//making our animation of ball work.

c.gridx--; //decreasing gridx by 1.

jTthree.setText("W"); //text set to W.

jBcompass.setIcon(icon\_pictures3); //setting new icon to compass.

jTtwo.setText(String.valueOf(nsq)); //new value set.

++nsq; //nsq increased by 1.

if(c.gridx == 0 && c.gridy ==12) {

jLgoal.setIcon(icon\_goldball); //making our animation of ball work.

JOptionPane.showMessageDialog(null,"Congratulation!! You Completed The Maze."); //displaying message box.

jLgoal.setIcon(icon\_sandstone);//making our animation of ball work.

jLball.setIcon(icon\_goldball);//making our animation of ball work.

c.gridx = 15; c.gridy = 0;

jTtwo.setText("1"); //set text to 1.

nsq = 1; //value of nsq is 1.

}

}

}

});

}

// method to check if ball can move toward right or not.

public boolean canMoveRight(int x) {

c.gridy = x; //gridx is equal to variable.

boolean canMoveFlag = false;//created a boolean variable.

if(c.gridy == 0 || c.gridy == 3 || c.gridy == 6 || c.gridy == 9 || c.gridy == 12 ){

canMoveFlag = true;//boolean value true.

}

return canMoveFlag; //boolean value returned.

}

//method created to run ball to end

public void runBall() {

int nval=Jruler.getValue()\*20; //getting value of Jruler.

run = new Timer(nval,new ActionListener() { //new timer function assigned.

@Override

public void actionPerformed(ActionEvent arg0) {

if((c.gridx==9 && c.gridy<3) || (c.gridx==6 && c.gridy<6) ||(c.gridx==5 && c.gridy<9)||(c.gridx==3 && c.gridy<12)) {

jLsands[c.gridx][c.gridy+1].setIcon(icon\_goldball);//making our animation of ball work.

jLsands[c.gridx][c.gridy].setIcon(icon\_sandroad);//making our animation of ball work.

c.gridy++; //gridx increased by 1.

jTthree.setText("S"); //text set to S

jBcompass.setIcon(icon\_pictures1); //new icon set on button.

jTtwo.setText(String.valueOf(nsq)); //new value for square textfield.

++nsq; //sq value increased by 1.

}

else {

jLball.setIcon(icon\_sandroad);//making our animation of ball work.

jLsands[c.gridx-1][c.gridy].setIcon(icon\_goldball);//making our animation of ball work.

jLsands[c.gridx][c.gridy].setIcon(icon\_sandroad);//making our animation of ball work.

c.gridx--; //decreasing value by 1.

jTthree.setText("W"); //setting new text.

jBcompass.setIcon(icon\_pictures3); //setting new icon.

jTtwo.setText(String.valueOf(nsq));

++nsq;

if(c.gridx == 0 && c.gridy ==12) {

jLgoal.setIcon(icon\_goldball); //making our animation of ball work.

JOptionPane.showMessageDialog(null,"Congratulation!! You Completed The Maze."); //message generated.

run.stop(); //stopping the timer.

jBpause.setVisible(false); //disabing pause button again.

jBrun.setVisible(true); //enabling pause button again.

c.gridx = 15;c.gridy=0; //setting value of gridx and grid y.

jLball.setIcon(icon\_goldball); // adding ball in its original position

jLgoal.setIcon(icon\_sandstone); //making animation for ball.

jBact.setEnabled(true); //act button enabled again.

digitalclock.stop(); //stopping the timer.

change.stop(); //stopping the timer.

}

jTthree.setText("W");//text set to W.

}}

});

jBpause.setVisible(true); //setting jBpause visible.

run.start(); //starting the timer.

}

//creating method for option3

public void makeChange() {

createGUI(); //createGUI(); called.

gc.gridy=12;

gc.gridx=1;

jLsands[gc.gridx][gc.gridy].setIcon(icon\_apple); //making animation for icon apple.

change = new Timer(600,new ActionListener() {

@Override

public void actionPerformed(ActionEvent arg0) {

if((gc.gridx==3 && gc.gridy>9 )||(gc.gridx==5 && gc.gridy>6) ||(gc.gridx==6 && gc.gridy>3)||(gc.gridx==9 &&gc.gridy>0)) {

jLsands[gc.gridx][gc.gridy-1].setIcon(icon\_apple);//making animation for icon apple.

jLsands[gc.gridx][gc.gridy].setIcon(icon\_sandroad);//making animation for icon apple.

gc.gridy--; //value decreased by 1.

}

else {

jLsands[gc.gridx+1][gc.gridy].setIcon(icon\_apple);//making animation for icon apple.

jLsands[gc.gridx][gc.gridy].setIcon(icon\_sandroad);//making animation for icon apple.

gc.gridx++; //value increased by 1.

}

if(gc.gridx==15) {

digitalclock.stop();//stopping digital timer.

change.stop();

JOptionPane.showMessageDialog(null, "Sorry !!! You Lose The Game!!!");//message box displayed.

run.stop();//stopping run timer.

createGUI(); //createGUI called.

}

}

});

change.start(); //change timer started.

}

// creating method for moving down

public void moveDown() throws InterruptedException {

Timer down = new Timer((Jruler.getValue()\*20), new ActionListener() { //down timer initialized.

@Override

public void actionPerformed(ActionEvent arg0) {

jLsands[c.gridx][c.gridy+1].setIcon(icon\_goldball); //making animation for ball.

jLsands[c.gridx][c.gridy].setIcon(icon\_sandroad);//making animation for ball.

playSound("fall.wav"); // playing sounnd by calling method playSound

c.gridy++; //value increased by 1.

jTthree.setText("S"); //setting text 1.

jBcompass.setIcon(icon\_pictures1); //setting icon on button.

jTtwo.setText(String.valueOf(nsq)); //new value set for textfield of square.

++nsq; //value increased by 1.

}

});

down.start(); // down timer started.

}

//creating method for moving up

public void moveUp() {

jLsands[c.gridx][c.gridy-1].setIcon(icon\_goldball);//making animation for ball.

jLsands[c.gridx][c.gridy].setIcon(icon\_sandroad);//making animation for ball.

c.gridy--;//value decreased by 1.

jTthree.setText("N"); //setting a new text.

}

//creating method for moving left

public void moveLeft() throws InterruptedException {

jLball.setIcon(icon\_sandroad);//making animation for ball.

jLsands[c.gridx-1][c.gridy].setIcon(icon\_goldball);//making animation for ball.

jLsands[c.gridx][c.gridy].setIcon(icon\_sandroad);//making animation for ball.

c.gridx--;//value decreased by 1.

if(c.gridx == 0 && c.gridy ==12) {

jLgoal.setIcon(icon\_goldball); //making animation for ball.

JOptionPane.showMessageDialog(null,"Congratulation!! You Completed The Maze.");//message dialog popped out.

c.gridx = 15;c.gridy=0; //setting value of gridx and grid y.

jLball.setIcon(icon\_goldball); // adding ball in its original position

jLgoal.setIcon(icon\_sandstone); //making animation for ball.

change.stop(); //change timer stopped.

digitalclock.stop();//digital timer stopped.

}

jTthree.setText("W"); //text set to W.

}

//creating method for moving right

public void moveRight() {

jLsands[c.gridx+1][c.gridy].setIcon(icon\_goldball);//making animation for ball.

jLsands[c.gridx][c.gridy].setIcon(icon\_sandroad);//making animation for ball.

c.gridx++; //increasing value by 1.

if(c.gridx == 1 && c.gridy ==12) {

jLgoal.setIcon(icon\_sandstone);//making animation for ball.

}

jTthree.setText("E"); //text Set to E.

}

//creating method for digitalclock.

public void digitalclock(JTextField sec,JTextField min) {

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

@Override

public void actionPerformed(ActionEvent arg0) {

sec.setText(Integer.toString(nticks / 60)); //setting text to sec textfield.

min.setText(Integer.toString(nticks % 60));//setting text to min textfield.

nticks = nticks + 1; //value to nticks increased by 1.

}} );

digitalclock.start(); //digital timer started.

}

Clip clip; //variable to store sound effect in java.

//importing sound in the java for movedown

public void playSound(String sound) { //method created for playing sound effect.

try {

sPlay(); //calling method sPlay();

AudioInputStream inStream = AudioSystem.getAudioInputStream(new File("sounds/" + sound)); //importing new audiostream.

clip = AudioSystem.getClip(); //assigning clip to audio stream.

clip.open(inStream); //opening the audio file inside audio stream.

clip.start(); //starting the clip.

}catch(Exception e){

e.printStackTrace();

sPlay(); //method sPlay() called.

}

}

private void sPlay() {

if(clip != null) {

clip.stop(); //stopping the clip.

clip.close(); //closing the clip.

clip = null; //making clip null.

}

}

//implementing Action performed when button clicked

@Override

public void actionPerformed(ActionEvent event) {

if (event.getActionCommand().equals("Exit")) {

System.exit(0); //exiting the system.

}

if(event.getActionCommand().equals("<")){

//making movement of the ball

try {

moveLeft(); //callling function moveleft();

jBcompass.setIcon(icon\_pictures3); //setting new icon.

jTtwo.setText(Integer.toString(nsq));//setting text to square textfield.

++nsq; //value iincreased by 1.

} catch (Exception e) {

e.printStackTrace();

}

}

if(event.getActionCommand().equals("^")){

try {

//making movement of the ball

moveUp() ; //calling function move up.

jBcompass.setIcon(icon\_pictures); //new icon set on compass.

jTtwo.setText(Integer.toString(nsq));//setting text to square textfield.

++nsq; //value increased by 1.

}

catch (Exception e) {

e.printStackTrace();

}

}

if(event.getActionCommand().equals(">")){

try {

//making movement of the ball

moveRight(); //calling function moveRight.

jBcompass.setIcon(icon\_pictures2); //setting new icon to compass.

jTtwo.setText(String.valueOf(nsq));//setting text to square textfield.

++nsq; //value increased by 1.

}

catch (Exception e) {

e.printStackTrace();

}

}

//making ball fall down automatically

try {

moveDown(); //calling function moveDown();

} catch (InterruptedException e) {

e.printStackTrace();

}

}

//making ball run in key press.

@Override

public void keyPressed(KeyEvent e) {

// TODO Auto-generated method stub

int key = e.getKeyCode();

if (key == KeyEvent.VK\_UP) {

moveUp(); //calling function moveUp.

}

else if ( key == KeyEvent.VK\_LEFT) {

try {

moveLeft(); //calling function moveLeft.

} catch (InterruptedException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

}

else if (key == KeyEvent.VK\_RIGHT) {

moveRight(); //calling function moveRight.

}

else if (key == KeyEvent.VK\_DOWN) {

try {

moveDown();//calling function moveDown.

} catch (InterruptedException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

}

}

@Override

public void keyReleased(KeyEvent arg0) {}

@Override

public void keyTyped(KeyEvent arg0) {}

}