



**City International School**  
CREATING GLOBAL CITIZENS

# INFORMATICS PRACTICES PROJECT

## 2016 - 2017

### ***GAME ZONE***

Welcome Omkar Nath

**HOME SCREEN**

Change Password  
Delete Account  
Sign Out

**REMEMBER THE SEQUENCE**

Click to Play !!

Your High Score: 0

**GET THE SQUARES**

Click to Play !!

Your High Score: 79

**MATCH THE COLOURS**

Click to Play !!

Your High Score: 173

**'Remember The Sequence' High Scores**

Rank	Username	Score
1	Shyam	135
2	mani	87
3	PhyLab	50
4	Samruddhi	10
5	ChemLab	10

**'Get The Squares' High Scores**

Rank	Username	Score
1	nirmal	103
2	Omkar Nath	79
3	ron	13
4	PhyLab	12
5	Shreya	8

**'Match The Colours' High Scores**

Rank	Username	Score
1	Omkar Nath	173
2	PhyLab	22
3	Shreya	21
4	nirmal	15
5	Samruddhi	9

Name: OMKAR SUNIL NATH

Class: XII

Roll Number:



# Informatics Practices

## Project Report

### ***"GAME ZONE"***

*2016 - 2017*

Prepared By

Name:

Class:

Roll No:



## CERTIFICATE

This is to certify that **Omkar Sunil Nath** of Class **XII** with Roll Number \_\_\_\_\_ has satisfactorily completed the project in **Informatics Practices** in partial fulfillment of the curriculum of the Central Board of Secondary Education, leading to the award for the All India Senior School Certificate Examination (**AISSCE**) for the academic year **2016 – 2017** and has submitted the project report in time.

\_\_\_\_\_  
**INTERNAL EXAMINER**

\_\_\_\_\_  
**EXTERNAL EXAMINER**

\_\_\_\_\_  
**PRINCIPAL**

\_\_\_\_\_  
**SCHOOL STAMP**

# ACKNOWLEDGEMENT

I would like to thank my informatics practices teacher, Mrs. Swati Shrivastava, for her teaching and guidance throughout the making of this project.

I would like to thank my principal, Mrs. Pooja Arora, for managing the school and providing an enriching learning environment.

I would also like to thank my parents for providing me with the necessary motivation and support.

# TABLE OF CONTENTS

Serial Number	Topic	Page Number
1.	Certificate	2
2.	Acknowledgement	3
3.	Table of Contents	4
4.	System Requirements	5
	Hardware Requirements	5
	Software Requirements	5
5.	About The Project	6-7
6.	Source Code and Screenshots	8-82
	Frame: Home Page	8-27
	Frame: Remember The Sequence (Game 1)	28-41
	Frame: Get The Squares (Game 2)	42-61
	Frame: Match The Colours (Game 3)	62-82
7.	MySQL Table Description	83
8.	Bibliography	84
9.	CD Containing The Program	85

# SYSTEM REQUIREMENTS

## HARDWARE REQUIRMENTS:

Storage Media CD: 700 MB

Memory: 512 MB

Hard Disk: 40 GB

Printer

## SOFTWARE REQUIREMENTS:

IDE: NetBeans v.6.9

Java: JDK v.6u20

Database: MySQL v.5.7

Operating System: Windows 7

Documentation: Microsoft Word 2010

# ABOUT THE PROJECT

“Game Zone” as the name suggests, is an application designed for gaming. It is a connectivity based program that has been created using java based IDE “NetBeans” for the graphical ‘front end’ and database, “MySQL” as the ‘back end’. The front end provides a simple, user-friendly, easy to use interactive interface, while using the back end for data storage and manipulation.

The application consists of three games, “Remember The Sequence”, “Get The Squares” and “Match The Colours”. A user first needs to create a new account with a unique username and password. The user can, using the account, play any of the three games. The highest score of the user account in every single game is stored and updated as required. Also, the top five scores in each game amongst all accounts are visible for all the users to see.

Using the application, the user is able to perform all three tasks of data manipulation which are: data insertion by creating an account, data manipulation by changing the account password, and data deletion by deleting the account. This all, apart from dynamically updating and accessing the high scores as and when is required. Furthermore, provision has also been made for an administrative account which can be used to access high score data of all accounts.

The games themselves are not too sophisticated in their graphics and use components provided by Netbeans. The functionality of these components has been vastly expanded by the use of code for handling numerous events. Various mathematical models like non-linear functions and arithmetic progressions have been used for score generation and time management. The games have been designed to provide for an ever increasing level of difficulty as one advances through the levels. Points are accordingly awarded based on the difficulty. This provides for a competitive gaming environment which has no limit to how far one can go.

The games are all memory based games, which challenge both the intellect and the attentiveness of the user. The implementations of the concepts of all three games are largely based on timer functions, which have been vastly incorporated into the application. In all the games, an interactive experience has been provided for by the use of various events like `MouseEntered`, `MouseExited` and `MouseClicked`. The use of

audio (sound effects) and visual tools (pictures and colours) have also greatly enhanced the gaming experience of all three games.

The first game titled 'Remember The Sequence', consists of four squares (text fields coloured Red, Green, Blue, Yellow). Some of these squares will light up in a random sequence. The user's task is to remember the sequence in which the coloured squares light up and click the text fields in the same sequence. As the user progresses through the levels, the length of the sequence and the points awarded increases. In the Nth level, the total length of the sequence is 'N'.

The second game titled 'Get The Squares' consists of a grid of sixteen squares (text fields). Out of these, some number (between 1 and 16) of squares will light up for a short time interval.. The user is supposed to remember and select all the squares that light up. As the user progresses, the difficulty increases by decreasing the time for which each set is shown. Points are awarded based on the number of squares shown, and the time for which they are shown.

The third game titled 'Match The Colours' consists of two columns of four squares (text fields) each. The colours red, green, blue and yellow are distributed in each column and displayed for a short time. The user is supposed to select a colour from one column, and select the same colour from the other column. Here also, difficulty is increased by decreasing the time interval for which the squares light up. Points are awarded based on the time for which the squares are shown, and the difficulty in selecting each pair of colours (In each set, the first two are the hardest and the last two are obvious).

The entire concept of the application was not to take complicated, intense and heavy on graphics games, but rather to take three games and create an interface around it at a level that would satisfy the criteria of a professional application. The application thus created can take care of almost any problem it might encounter. Confirmation dialog boxes of JOptionPane have been shown as a cautionary message whenever required. Scores are updated in the database as soon as a point is awarded, rather than waiting till the game gets over. The top five scores are checked and updated dynamically. Most importantly, the application interface has been kept very simple and easy to understand. There are prompts provided at every step of the way to help the user understand what to do next, while keeping the information supply to the user at a minimalistic level. This, combined with the use of audio and visual tools to interact with the user provides the user a comprehensive gaming experience.



# SOURCE CODE AND SCREENSHOTS

## **Frame: Home Page**

### Importing Java Packages:

```
import com.mysql.jdbc.Connection;
import com.mysql.jdbc.Statement;
import java.sql.DriverManager;
import java.sql.ResultSet;

import java.awt.Toolkit;
import java.io.File;
import java.io.FileInputStream;
import java.io.InputStream;
import sun.audio.AudioPlayer;
import sun.audio.AudioStream;

import java.util.*;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;
```

### Global Variables:

```
Timer timerGameMsgShow;
Timer timerGameMsgHide;

boolean oneTime = true;
boolean begin1 = true;
boolean begin2 = true;

String username = "";

int msGameMsgFlash = 1000;

int game1HS;
int game2HS;
int game3HS;
int rowsG1 = 0;
int rowsG2 = 0;
int rowsG3 = 0;

int rowsAT = 0;
```

## Event – Form Window Activated:

```
private void formWindowActivated(java.awt.event.WindowEvent evt) {  
    if (oneTime) {  
        oneTime = false;  
  
        timerGameMsgShow = new Timer();  
        timerGameMsgHide = new Timer();  
  
        timerGameMsgShow.scheduleAtFixedRate(new taskGameMsgShow(), 0, msGameMsgFlash);  
        timerGameMsgHide.scheduleAtFixedRate(new taskGameMsgHide(), msGameMsgFlash / 2,  
msGameMsgFlash);  
    }  
  
    if (begin1 && username.isEmpty()) {  
        detailsPanel.setVisible(false);  
        homeScreenPanel.setVisible(false);  
        adminPanel.setVisible(false);  
        accountPanel.setVisible(false);  
        game1Panel.setVisible(false);  
        game2Panel.setVisible(false);  
        game3Panel.setVisible(false);  
        coverPanel.setVisible(false);  
  
        gameZonePanel.setVisible(true);  
        game1HSPanel.setVisible(true);  
        game2HSPanel.setVisible(true);  
        game3HSPanel.setVisible(true);  
  
        initial(true);  
  
        begin1 = false;  
    } else if (begin1 && !username.equals("Admin")) {  
        detailsPanel.setVisible(false);  
        adminPanel.setVisible(false);  
        accountPanel.setVisible(false);  
        gameZonePanel.setVisible(false);  
  
        initial(false);  
  
        homeScreenPanel.setVisible(true);  
        game1Panel.setVisible(true);  
        game2Panel.setVisible(true);  
        game3Panel.setVisible(true);  
        game1HSPanel.setVisible(true);  
        game2HSPanel.setVisible(true);  
        game3HSPanel.setVisible(true);  
  
        coverPanel.setVisible(false);  
  
        begin1 = false;  
    }  
}
```

```
if (begin2) {
    begin2 = false;

    while (rowsG1 > 0) {
        ((DefaultTableModel) game1Table.getModel()).removeRow(0);
        rowsG1--;
    }
    while (rowsG2 > 0) {
        ((DefaultTableModel) game2Table.getModel()).removeRow(0);
        rowsG2--;
    }
    while (rowsG3 > 0) {
        ((DefaultTableModel) game3Table.getModel()).removeRow(0);
        rowsG3--;
    }

    try {
        Class.forName("java.sql.DriverManager");
        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
        Statement stmt = (Statement) con.createStatement();
        String query = "SELECT username,rememberthesequence FROM highscores ORDER BY
rememberthesequence DESC";
        ResultSet rs = stmt.executeQuery(query);
        int i = 1;
        boolean loop = true;
        while (loop && i <= 5) {
            if (rs.next()) {
                DefaultTableModel model = (DefaultTableModel) game1Table.getModel();
                model.addRow(new Object[] {Integer.toString(i), rs.getString("username"),
rs.getString("rememberthesequence")});
                rowsG1++;
                i++;
            } else {
                loop = false;
            }
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }

    try {
        Class.forName("java.sql.DriverManager");
        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
        Statement stmt = (Statement) con.createStatement();
        String query = "SELECT username,getthesquares FROM highscores ORDER BY getthesquares
DESC";
        ResultSet rs = stmt.executeQuery(query);
        int i = 1;
        boolean loop = true;
        while (loop && i <= 5) {
            if (rs.next()) {
                DefaultTableModel model = (DefaultTableModel) game2Table.getModel();
```

```

        model.addRow(new Object[]{Integer.toString(i), rs.getString("username"),
rs.getString("getthesquares")});
        rowsG2++;
        i++;
    } else {
        loop = false;
    }
}
} catch (Exception e) {
    JOptionPane.showMessageDialog(this, e.getMessage());
}

try {
    Class.forName("java.sql.DriverManager");
    Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
    Statement stmt = (Statement) con.createStatement();
    String query = "SELECT username,matchthecolours FROM highscores ORDER BY
matchthecolours DESC";
    ResultSet rs = stmt.executeQuery(query);
    int i = 1;
    boolean loop = true;
    while (loop && i <= 5) {
        if (rs.next()) {
            DefaultTableModel model = (DefaultTableModel) game3Table.getModel();
            model.addRow(new Object[]{Integer.toString(i), rs.getString("username"),
rs.getString("matchthecolours")});
            rowsG3++;
            i++;
        } else {
            loop = false;
        }
    }
} catch (Exception e) {
    JOptionPane.showMessageDialog(this, e.getMessage());
}
}
}

```

## Methods & Functions Used:

```

class taskGameMsgShow extends TimerTask {
    public void run() {
        game1PlayL.setVisible(true);
        game2PlayL.setVisible(true);
        game3PlayL.setVisible(true);
    }
}

```

```

class taskGameMsgHide extends TimerTask {
    public void run() {
        game1PlayL.setVisible(false);
        game2PlayL.setVisible(false);
        game3PlayL.setVisible(false);
    }
}

```

```
    }  
}  
  
public void initial(boolean state) {  
    signIn1Btn.setVisible(state);  
    newUserBtn.setVisible(state);  
    exit1Btn.setVisible(state);  
}  
  
public void visible1() {  
    usernameTF.setText("");  
    currentPasswordPF.setText("");  
    passwordPF.setText("");  
    confirmPasswordPF.setText("");  
  
    currentPasswordL.setVisible(false);  
    newPasswordL.setVisible(false);  
    confirmNewPasswordL.setVisible(false);  
  
    currentPasswordPF.setVisible(false);  
  
    resetBtn.setVisible(false);  
    cancel1Btn.setVisible(false);  
    cancel2Btn.setVisible(false);  
    deleteBtn.setVisible(false);  
  
    usernameL.setVisible(true);  
    characterLimitL.setVisible(true);  
    passwordL.setVisible(true);  
    confirmPasswordL.setVisible(true);  
  
    usernameTF.setVisible(true);  
  
    passwordPF.setVisible(true);  
    confirmPasswordPF.setVisible(true);  
  
    signIn2Btn.setVisible(true);  
    createAccountBtn.setVisible(true);  
    back2ABtn.setVisible(true);  
    back2BBtn.setVisible(true);  
}  
  
public void visible2() {  
    usernameTF.setText("");  
    currentPasswordPF.setText("");  
    passwordPF.setText("");  
    confirmPasswordPF.setText("");  
  
    usernameL.setVisible(false);  
    characterLimitL.setVisible(false);  
    passwordL.setVisible(false);  
    confirmPasswordL.setVisible(false);  
  
    usernameTF.setVisible(false);
```

```
signIn2Btn.setVisible(false);
createAccountBtn.setVisible(false);
back2ABtn.setVisible(false);
back2BBtn.setVisible(false);
cancel2Btn.setVisible(false);
deleteBtn.setVisible(false);

currentPasswordL.setVisible(true);
newPasswordL.setVisible(true);
confirmNewPasswordL.setVisible(true);

passwordPF.setVisible(true);
currentPasswordPF.setVisible(true);
confirmPasswordPF.setVisible(true);

resetBtn.setVisible(true);
cancel1Btn.setVisible(true);
}

public void visible3() {
    usernameTF.setText("");
    currentPasswordPF.setText("");
    passwordPF.setText("");
    confirmPasswordPF.setText("");

    usernameL.setVisible(false);
    characterLimitL.setVisible(false);
    currentPasswordL.setVisible(false);
    confirmPasswordL.setVisible(false);
    newPasswordL.setVisible(false);
    confirmNewPasswordL.setVisible(false);

    usernameTF.setVisible(false);

    currentPasswordPF.setVisible(false);
    confirmPasswordPF.setVisible(false);

    signIn2Btn.setVisible(false);
    createAccountBtn.setVisible(false);
    back2ABtn.setVisible(false);
    back2BBtn.setVisible(false);
    resetBtn.setVisible(false);
    cancel1Btn.setVisible(false);

    passwordL.setVisible(true);

    passwordPF.setVisible(true);

    deleteBtn.setVisible(true);
    cancel2Btn.setVisible(true);
}

public void dataPass0(String uName) {
    username = uName;
    try {
```

```
visible1();
detailsPanel.setVisible(false);
gameZonePanel.setVisible(false);

Class.forName("java.sql.DriverManager");
Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
Statement stmt = (Statement) con.createStatement();
String query = "SELECT * FROM highscores WHERE Username = '" + uName + "'";
ResultSet rs = stmt.executeQuery(query);
if (rs.next()) {
    game1HS = rs.getInt("RememberTheSequence");
    game2HS = rs.getInt("getthesquares");
    game3HS = rs.getInt("matchthecolours");
}

nameL.setText(username);
HS1L.setText("" + game1HS);
HS2L.setText("" + game2HS);
HS3L.setText("" + game3HS);

game1Panel.setVisible(true);
game2Panel.setVisible(true);
game3Panel.setVisible(true);
homeScreenPanel.setVisible(true);

} catch (Exception e) {
    JOptionPane.showMessageDialog(this, e.getMessage());
}
}

public void playAudio(int track) {
    switch (track) {
        case 0:
            Toolkit.getDefaultToolkit().beep();
            break;
        case 1:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\proceedToGame.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 2:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\singleBeep.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 3:
```

```
        try {
            InputStream music = new FileInputStream(new File("src\\Audio\\glassPing.wav"));
            AudioStream audio = new AudioStream(music);
            AudioPlayer.player.start(audio);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
        break;
    case 4:
        try {
            InputStream music = new FileInputStream(new File("src\\Audio\\singleCoinDrop.wav"));
            AudioStream audio = new AudioStream(music);
            AudioPlayer.player.start(audio);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
        break;
    case 5:
        try {
            InputStream music = new FileInputStream(new File("src\\Audio\\multipleCoinDrop.wav"));
            AudioStream audio = new AudioStream(music);
            AudioPlayer.player.start(audio);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
        break;
    case 6:
        try {
            InputStream music = new FileInputStream(new File("src\\Audio\\tadaHighScore.wav"));
            AudioStream audio = new AudioStream(music);
            AudioPlayer.player.start(audio);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
        break;
    case 7:
        try {
            InputStream music = new FileInputStream(new File("src\\Audio\\buzzerGameOver.wav"));
            AudioStream audio = new AudioStream(music);
            AudioPlayer.player.start(audio);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
        break;
    }
}
```



## Home Screen (Program Start):

### Button – Sign In:

```
private void signInBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    initial(false);  
  
    detailsPanel.setVisible(true);  
  
    visible1();  
    characterLimitL.setVisible(false);  
    confirmPasswordL.setVisible(false);  
    confirmPasswordPF.setVisible(false);  
    createAccountBtn.setVisible(false);  
    back2BBtn.setVisible(false);  
}
```

### Button – New User:

```
private void newUserBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    initial(false);  
  
    detailsPanel.setVisible(true);  
  
    visible1();  
    signIn2Btn.setVisible(false);  
    back2ABtn.setVisible(false);  
}
```

### Button – Exit:

```
private void exit1BtnActionPerformed(java.awt.event.ActionEvent evt) {  
    System.exit(0);  
}
```

## Create Account:

### Button – Back:

```
private void back2BBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    detailsPanel.setVisible(false);  
  
    initial(true);  
}
```

### Button – Create Account:

```
private void createAccountBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    boolean flag = true;  
    boolean loop = true;  
    String uName = usernameTF.getText();  
    String pWord1 = new String(passwordPF.getPassword());
```

**WELCOME TO THE GAME ZONE**

**SIGN IN**      **NEW USER**

**EXIT**

**'Remember The Sequence' High Scores**

Rank	Username	Score
1	Shyam	135
2	mani	87
3	PhyLab	50
4	Samruddhi	10
5	ChemLab	10

**'Get The Squares' High Scores**

Rank	Username	Score
1	nirmal	103
2	ron	13
3	PhyLab	12
4	Shreya	8
5	Samruddhi	4

**'Match The Colours' High Scores**

Rank	Username	Score
1	PhyLab	22
2	Shreya	21
3	nirmal	15
4	Samruddhi	9
5	ChemLab	7

Program Start Home Screen

**WELCOME TO THE GAME ZONE**

Username (Upto 10 characters)

Password

Confirm Password

**Back**      **Create Account**

**'Remember The Sequence' High Scores**

Rank	Username	Score
1	Shyam	135
2	mani	87
3	PhyLab	50
4	Samruddhi	10
5	ChemLab	10

**'Get The Squares' High Scores**

Rank	Username	Score
1	nirmal	103
2	ron	13
3	PhyLab	12
4	Shreya	8
5	Samruddhi	4

**'Match The Colours' High Scores**

Rank	Username	Score
1	PhyLab	22
2	Shreya	21
3	nirmal	15
4	Samruddhi	9
5	ChemLab	7

Creating a User Account

```

String pWord2 = new String(confirmPasswordPF.getPassword());
String errorMessage = "";

if (uName.isEmpty()) {
    errorMessage = "Please enter the Username !!";
    flag = false;
} else if (pWord1.isEmpty()) {
    errorMessage = "Please enter the Password !!";
    flag = false;
} else if (pWord2.isEmpty()) {
    errorMessage = "Please confirm the Password !!";
    flag = false;
} else if (!(pWord1.equals(pWord2))) {
    errorMessage = "Passwords do not match !!";
    flag = false;
} else if (uName.length() > 10) {
    errorMessage = "Username should not be more than 10 characters long";
    flag = false;
} else {
    try {
        Class.forName("java.sql.DriverManager");
        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
        Statement stmt = (Statement) con.createStatement();
        String query = "SELECT username FROM userAccounts";
        ResultSet rs = stmt.executeQuery(query);
        while (loop) {
            if (rs.next()) {
                if (uName.toLowerCase().equals(rs.getString("Username").toLowerCase())) {
                    flag = false;
                    loop = false;
                    errorMessage = "Username already exists !! Please try another username.";
                }
            } else {
                loop = false;
            }
        }
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
        visible1();
        signIn2Btn.setVisible(false);
        back2ABtn.setVisible(false);
    }
}

if (flag) {
    try {
        Class.forName("java.sql.DriverManager");
        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
        Statement stmt = (Statement) con.createStatement();
        String query1 = "INSERT INTO userAccounts VALUES ('" + uName + "','" + pWord1 + "')";
        String query2 = "INSERT INTO highscores VALUES ('" + uName + "',0,0,0)";
        stmt.executeUpdate(query1);
        stmt.executeUpdate(query2);
    }
}

```

```

        begin2 = true;

        playAudio(0);
        JOptionPane.showMessageDialog(this, "Account created Successfully!!");

        visible1();
        detailsPanel.setVisible(false);
        initial(true);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
        visible1();
        signIn2Btn.setVisible(false);
        back2ABtn.setVisible(false);
    }
} else if (!flag) {
    playAudio(0);
    JOptionPane.showMessageDialog(this, errorMessage);
    visible1();
    signIn2Btn.setVisible(false);
    back2ABtn.setVisible(false);
}
}

```

### Sign In:

#### Button – Back:

```

private void back2ABtnActionPerformed(java.awt.event.ActionEvent evt) {
    detailsPanel.setVisible(false);

    initial(true);
}

```

#### Button – Sign In:

```

private void signIn2BtnActionPerformed(java.awt.event.ActionEvent evt) {
    boolean flag = true;
    String uName = usernameTF.getText();
    String pWord = new String(passwordPF.getPassword());
    String errorMessage = "";

    if (uName.isEmpty()) {
        errorMessage = "Please enter the Username !!";
        flag = false;
    } else if (pWord.isEmpty()) {
        errorMessage = "Please enter the Password !!";
        flag = false;
    } else if (uName.length() > 10) {
        errorMessage = "Username should not be more than 10 characters long";
        flag = false;
    } else {
        boolean loop = true;
        boolean flag2 = true;
    }
}

```

**WELCOME TO THE GAME ZONE**

Username

Password

**'Remember The Sequence' High Scores**

Rank	Username	Score
1	Shyam	135
2	mani	87
3	PhyLab	50
4	Samruddhi	10
5	ChemLab	10

**'Get The Squares' High Scores**

Rank	Username	Score
1	nirmal	103
2	iron	13
3	PhyLab	12
4	Shreya	8
5	Samruddhi	4

**'Match The Colours' High Scores**

Rank	Username	Score
1	PhyLab	22
2	Shreya	21
3	nirmal	15
4	Samruddhi	9
5	ChemLab	7

Signing into the User Account

```
try {
    Class.forName("java.sql.DriverManager");
    Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
    Statement stmt = (Statement) con.createStatement();
    String query = "SELECT username FROM userAccounts";
    ResultSet rs = stmt.executeQuery(query);
    while (loop) {
        if (rs.next()) {
            if (uName.equals(rs.getString("Username"))) {
                flag2 = false;
                loop = false;
            }
        } else {
            loop = false;
        }
    }
    if (flag2) {
        flag = false;
        errorMessage = "Username does not exist !!";
    }
} catch (Exception e) {
    JOptionPane.showMessageDialog(this, e.getMessage());
    visible1();
    signIn2Btn.setVisible(false);
    back2ABtn.setVisible(false);
}

if (flag) {
    try {
        Class.forName("java.sql.DriverManager");
        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
        Statement stmt = (Statement) con.createStatement();
        String query1 = "SELECT * FROM userAccounts WHERE username = '" + uName + "'";
        ResultSet rs1 = stmt.executeQuery(query1);
        if (rs1.next()) {
            if (pWord.equals(rs1.getString("Password"))) {
                if (uName.equals("Admin")) {
                    visible1();
                    detailsPanel.setVisible(false);
                    gameZonePanel.setVisible(false);

                    username = uName;

                    boolean loop = true;

                    String query2 = "SELECT * FROM highscores ORDER BY Username";
                    ResultSet rs2 = stmt.executeQuery(query2);

                    while (rowsAT > 0) {
                        ((DefaultTableModel) accountTable.getModel()).removeRow(0);
                        rowsAT--;
```

```

    }

    while (loop) {
        if (rs2.next()) {
            DefaultTableModel model = (DefaultTableModel) accountTable.getModel();
            model.addRow(new Object[]{rs2.getString("username"),
rs2.getString("rememberthesequence"), rs2.getString("getthesquares"), rs2.getString("matchthecolours")});
            rowsAT++;
        } else {
            loop = false;
        }
    }

    adminPanel.setVisible(true);
    accountPanel.setVisible(true);
} else {
    visible1();
    detailsPanel.setVisible(false);
    gameZonePanel.setVisible(false);

    username = uName;

    String query2 = "SELECT * FROM highscores WHERE Username = '" + uName + "'";
    ResultSet rs2 = stmt.executeQuery(query2);
    if (rs2.next()) {
        game1HS = rs2.getInt("RememberTheSequence");
        game2HS = rs2.getInt("getthesquares");
        game3HS = rs2.getInt("matchthecolours");
    }

    nameL.setText(username);
    HS1L.setText("" + game1HS);
    HS2L.setText("" + game2HS);
    HS3L.setText("" + game3HS);

    game1Panel.setVisible(true);
    game2Panel.setVisible(true);
    game3Panel.setVisible(true);
    homeScreenPanel.setVisible(true);
}
} else {
    playAudio(0);
    JOptionPane.showMessageDialog(this, "Please enter the correct password !!");
    visible1();
    characterLimitL.setVisible(false);
    confirmPasswordL.setVisible(false);
    confirmPasswordPF.setVisible(false);
    createAccountBtn.setVisible(false);
    back2BBtn.setVisible(false);
}
} else {
    playAudio(0);
    JOptionPane.showMessageDialog(this, "Username does not exist !!");
    visible1();
    characterLimitL.setVisible(false);

```

```

        confirmPasswordL.setVisible(false);
        confirmPasswordPF.setVisible(false);
        createAccountBtn.setVisible(false);
        back2BBtn.setVisible(false);
    }
} catch (Exception e) {
    JOptionPane.showMessageDialog(this, e.getMessage());
    visible1();
    characterLimitL.setVisible(false);
    confirmPasswordL.setVisible(false);
    confirmPasswordPF.setVisible(false);
    createAccountBtn.setVisible(false);
    back2BBtn.setVisible(false);
}
} else if (!flag) {
    playAudio(0);
    JOptionPane.showMessageDialog(this, errorMessage);
    visible1();
    characterLimitL.setVisible(false);
    confirmPasswordL.setVisible(false);
    confirmPasswordPF.setVisible(false);
    createAccountBtn.setVisible(false);
    back2BBtn.setVisible(false);
}
}

```

## User Account Logged In Home Screen:

### Button – Sign Out:

```

private void signOutBtnActionPerformed(java.awt.event.ActionEvent evt) {
    game1Panel.setVisible(false);
    game2Panel.setVisible(false);
    game3Panel.setVisible(false);
    homeScreenPanel.setVisible(false);

    initial(true);

    gameZonePanel.setVisible(true);

    username = "";
    game1HS = 0;
    game2HS = 0;
    game3HS = 0;
}

```

### Button – Change Password:

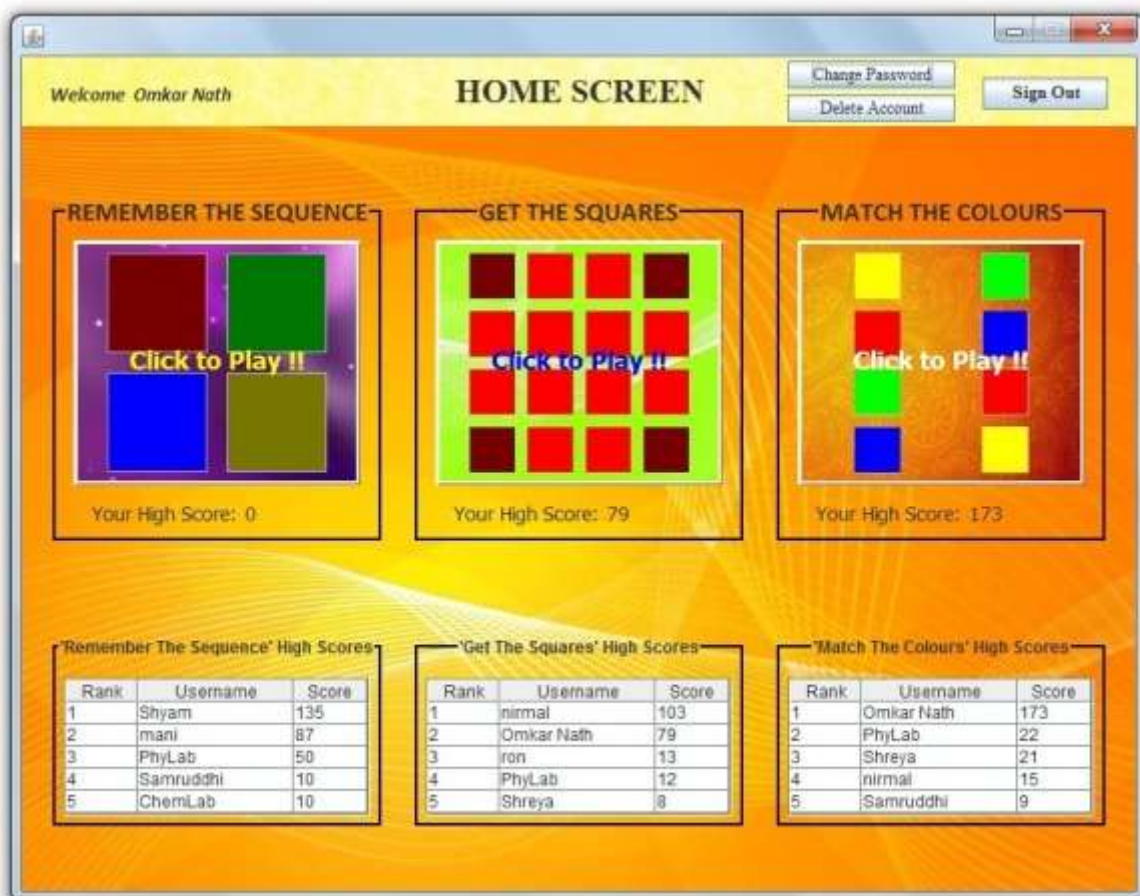
```

private void changePasswordBtnActionPerformed(java.awt.event.ActionEvent evt) {
    game1Panel.setVisible(false);
    game2Panel.setVisible(false);
    game3Panel.setVisible(false);

    signOutBtn.setVisible(false);
}

```





Home Screen of User Account after Logging In

```
changePasswordBtn.setVisible(false);
deleteAccountBtn.setVisible(false);

visible2();

detailsPanel.setVisible(true);
}
```

### Button – Delete Account:

```
private void deleteAccountBtnActionPerformed(java.awt.event.ActionEvent evt) {
    game1Panel.setVisible(false);
    game2Panel.setVisible(false);
    game3Panel.setVisible(false);

    signOutBtn.setVisible(false);
    changePasswordBtn.setVisible(false);
    deleteAccountBtn.setVisible(false);

    visible3();

    detailsPanel.setVisible(true);
}
```

### Label – Game 1 (Remember the Sequence):

```
private void game1LMouseClicked(java.awt.event.MouseEvent evt) {
    rememberTheSequence frame1 = new rememberTheSequence();
    frame1.dataPass1(username, game1HS);
    homePage.this.setVisible(false);
    frame1.setVisible(true);
}
```

### Label – Game 2 (Get the Squares):

```
private void game2LMouseClicked(java.awt.event.MouseEvent evt) {
    getTheSquares frame2 = new getTheSquares();
    frame2.dataPass2(username, game2HS);
    homePage.this.setVisible(false);
    frame2.setVisible(true);
}
```

### Label – Game 3 (Match the Colours):

```
private void game3LMouseClicked(java.awt.event.MouseEvent evt) {
    matchTheColours frame3 = new matchTheColours();
    frame3.dataPass3(username, game3HS);
    homePage.this.setVisible(false);
    frame3.setVisible(true);
}
```

## Change Password:

### Button – Cancel:

```
private void cancelBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    detailsPanel.setVisible(false);  
  
    visible1();  
  
    game1Panel.setVisible(true);  
    game2Panel.setVisible(true);  
    game3Panel.setVisible(true);  
  
    if (username.equals("Admin")) {  
        game1Panel.setVisible(false);  
        game2Panel.setVisible(false);  
        game3Panel.setVisible(false);  
  
        accountPanel.setVisible(true);  
        adminSignOutBtn.setVisible(true);  
        adminChangePasswordBtn.setVisible(true);  
    }  
  
    signOutBtn.setVisible(true);  
    changePasswordBtn.setVisible(true);  
    deleteAccountBtn.setVisible(true);  
}
```

### Button – Reset:

```
private void resetBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    String pWord1 = new String(currentPasswordPF.getPassword());  
    String pWord2 = new String(passwordPF.getPassword());  
    String pWord3 = new String(confirmPasswordPF.getPassword());  
  
    boolean flag = false;  
    String errorMessage = "";  
    if (pWord1.isEmpty()) {  
        errorMessage = "Please enter your current password !!";  
        flag = true;  
    } else if (pWord2.isEmpty()) {  
        errorMessage = "Please enter the new password !!";  
        flag = true;  
    } else if (pWord3.isEmpty()) {  
        errorMessage = "Please confirm the new Password !!";  
        flag = true;  
    } else if (!(pWord2.equals(pWord3))) {  
        errorMessage = "Passwords do not match !!";  
        flag = true;  
    } else {  
        try {  
            Class.forName("java.sql.DriverManager");  
            Connection con = (Connection)  
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
```

Welcome: Omkar Nath

## HOME SCREEN

Current Password:

New Password:

Confirm New Password:

**'Remember The Sequence' High Scores**

Rank	Username	Score
1	Shyam	135
2	mani	87
3	PhyLab	50
4	Samruddhi	10
5	ChemLab	10

**'Get The Squares' High Scores**

Rank	Username	Score
1	nirmal	103
2	Omkar Nath	79
3	ron	13
4	PhyLab	12
5	Shreya	8

**'Match The Colours' High Scores**

Rank	Username	Score
1	Omkar Nath	173
2	PhyLab	22
3	Shreya	21
4	nirmal	15
5	Samruddhi	9

Changing the Account Password

```
Statement stmt = (Statement) con.createStatement();
String query1 = "SELECT * FROM UserAccounts WHERE Username = '" + username + "'";
ResultSet rs = stmt.executeQuery(query1);
if (rs.next()) {
    if (pWord1.equals(rs.getString("Password"))) {
        String query2 = "UPDATE UserAccounts SET Password = '" + pWord2 + "' WHERE
Username = '" + username + "'";
        stmt.executeUpdate(query2);

        playAudio(0);
        JOptionPane.showMessageDialog(this, "Password Updated Successfully");

        detailsPanel.setVisible(false);

        visible1();

        if (username.equals("Admin")) {
            game1Panel.setVisible(false);
            game2Panel.setVisible(false);
            game3Panel.setVisible(false);

            accountPanel.setVisible(true);
            adminSignOutBtn.setVisible(true);
            adminChangePasswordBtn.setVisible(true);
        } else {
            game1Panel.setVisible(true);
            game2Panel.setVisible(true);
            game3Panel.setVisible(true);

            signOutBtn.setVisible(true);
            changePasswordBtn.setVisible(true);
            deleteAccountBtn.setVisible(true);
        }
    } else {
        flag = true;
        errorMessage = "Please enter your current password correctly !!";
    }
}
} catch (Exception e) {
    JOptionPane.showMessageDialog(this, e.getMessage());
}

if (flag) {
    playAudio(0);
    JOptionPane.showMessageDialog(this, errorMessage);
    visible2();
}
}
```

## Delete User Account:

### Button – Cancel:

```
private void cancel2BtnActionPerformed(java.awt.event.ActionEvent evt) {
    detailsPanel.setVisible(false);

    visible1();

    game1Panel.setVisible(true);
    game2Panel.setVisible(true);
    game3Panel.setVisible(true);

    signOutBtn.setVisible(true);
    changePasswordBtn.setVisible(true);
    deleteAccountBtn.setVisible(true);
}
```

### Button – Delete:

```
private void deleteBtnActionPerformed(java.awt.event.ActionEvent evt) {
    String pWord = new String(passwordPF.getPassword());

    boolean flag = false;

    String errorMessage = "";

    if (pWord.isEmpty()) {
        errorMessage = "Please Enter The password";
        flag = true;
    } else {
        try {
            Class.forName("java.sql.DriverManager");
            Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
            Statement stmt = (Statement) con.createStatement();
            String query1 = "SELECT * FROM userAccounts WHERE Username = " + username + """;
            ResultSet rs = stmt.executeQuery(query1);

            if (rs.next()) {
                if (pWord.equals(rs.getString("Password"))) {
                    begin2 = true;

                    playAudio(0);
                    int reply = JOptionPane.showConfirmDialog(this, "All highscore data and account information
of \"\" + username + "\" will be lost.\nDo you wish to proceed ?", "Caution !!",
JOptionPane.YES_NO_OPTION);
                    if (reply == JOptionPane.YES_OPTION) {
                        String query2 = "DELETE FROM userAccounts WHERE Username = " + username + """;
                        String query3 = "DELETE FROM highscores WHERE Username = " + username + """;
                        stmt.executeUpdate(query2);
                        stmt.executeUpdate(query3);

                        playAudio(0);
                    }
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```



Entering Password to check User before Deleting Account



Final Confirmation before deleting Account and related Data

```
JOptionPane.showMessageDialog(this, "Account deleted sucessfully !!");

visible1();

homeScreenPanel.setVisible(false);
detailsPanel.setVisible(false);

signOutBtn.setVisible(true);
changePasswordBtn.setVisible(true);
deleteAccountBtn.setVisible(true);

initial(true);

gameZonePanel.setVisible(true);

username = "";
game1HS = 0;
game2HS = 0;
game3HS = 0;

} else {
    detailsPanel.setVisible(false);

    visible1();

    game1Panel.setVisible(true);
    game2Panel.setVisible(true);
    game3Panel.setVisible(true);

    signOutBtn.setVisible(true);
    changePasswordBtn.setVisible(true);
    deleteAccountBtn.setVisible(true);
}
} else {
    flag = true;
    errorMessage = "Please enter your current password correctly !!";
}
}
} catch (Exception e) {
    JOptionPane.showMessageDialog(this, e.getMessage());
}
}

if (flag) {
    playAudio(0);
    JOptionPane.showMessageDialog(this, errorMessage);
    visible3();
}
}
```



## Administrator Control:

### Button – Sign Out:

```
private void adminSignOutBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    adminPanel.setVisible(false);  
    accountPanel.setVisible(false);  
  
    initial(true);  
  
    gameZonePanel.setVisible(true);  
}
```

### Button – Change Password:

```
private void adminChangePasswordBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    accountPanel.setVisible(false);  
  
    adminSignOutBtn.setVisible(false);  
    adminChangePasswordBtn.setVisible(false);  
  
    visible2();  
  
    detailsPanel.setVisible(true);  
}
```



Administrator Screen

## **Frame: Remember the Sequence (Game 1)**

### **Importing Java Packages:**

```
import com.mysql.jdbc.Connection;  
import com.mysql.jdbc.Statement;  
import java.sql.DriverManager;
```

```
import java.awt.*;
```

```
import java.io.File;  
import java.io.FileInputStream;  
import java.io.InputStream;  
import sun.audio.AudioPlayer;  
import sun.audio.AudioStream;
```

```
import java.util.*;
```

```
import javax.swing.JOptionPane;
```

### **Global Variables:**

```
Timer timerDelay;  
Timer timerFlash;  
Timer timerCorrect;
```

```
int msTimeBegin = 250;  
int msTimeEnd = 1500;  
int msTimeFlash = 250;
```

```
boolean clickable;  
boolean newHighScore;  
boolean begin = true;  
boolean end = false;  
boolean exit = true;
```

```
int gameScore;  
int highScore;  
int gameLevel;  
int currentLevel;  
int sequence[];
```

```
String username = "";
```

## Event – Form Window Activated:

```
private void formWindowActivated(java.awt.event.WindowEvent evt) {  
    if (begin) {  
        if (highScore == 0) {  
            newHighScore = true;  
        } else {  
            newHighScore = false;  
        }  
  
        clickable = false;  
        begin = false;  
        exit = true;  
  
        gameLevel = 1;  
  
        gameScore = 0;  
        gameScoreL2.setText(Integer.toString(gameScore));  
  
        correctMsgL.setVisible(false);  
        submitMsg1L.setVisible(false);  
        submitMsg2L.setVisible(false);  
  
        startMsg1L.setVisible(true);  
        startMsg2L.setVisible(true);  
  
        gamePanel.setVisible(false);  
        gameOverPanel.setVisible(false);  
  
        coverPanel.setVisible(false);  
    }  
  
    setBoxNormal(1, 1, 1, 1);  
}
```

## Methods & Functions Used:

```
public void playSequence(int msDelay, int msFlash) {  
    currentLevel = 1;  
    for (int i = 1; i <= gameLevel; i++) {  
        timerDelay = new Timer();  
        timerFlash = new Timer();  
        timerDelay.schedule(new taskDelay(), ((msDelay + msFlash) * (i - 1)) + msDelay);  
        timerFlash.schedule(new taskFlash(), ((msDelay + msFlash) * (i - 1)) + msDelay + msFlash);  
    }  
}  
  
class taskDelay extends TimerTask {  
    public void run() {  
        playAudio(2);  
        switch (sequence[currentLevel]) {  
            case 1:  
                setBoxHighlight(1, 0, 0, 0);  
                break;  
        }  
    }  
}
```

```
        case 2:
            setBoxHighlight(0, 1, 0, 0);
            break;
        case 3:
            setBoxHighlight(0, 0, 1, 0);
            break;
        case 4:
            setBoxHighlight(0, 0, 0, 1);
            break;
    }
}
}

class taskFlash extends TimerTask {
    public void run() {
        setBoxNormal(1, 1, 1, 1);
        if (++currentLevel > gameLevel) {
            submitMsg1L.setVisible(true);
            submitMsg2L.setVisible(true);

            clickable = true;
            currentLevel = 1;
        }
    }
}

public void newLevel() {
    clickable = false;

    gameLevel++;

    submitMsg1L.setVisible(false);
    submitMsg2L.setVisible(false);
    correctMsgL.setVisible(true);

    setBoxNormal(1, 1, 1, 1);

    nextLevel(msTimeEnd);
}

public void nextLevel(int msTime) {
    timerCorrect = new Timer();
    timerCorrect.schedule(new taskCorrect(), msTime);
}

class taskCorrect extends TimerTask {
    public void run() {
        correctMsgL.setVisible(false);

        startBtn.setText("Start Level " + gameLevel);
        startBtn.setVisible(true);

        startMsg1L.setVisible(true);
        startMsg2L.setVisible(true);
    }
}
```

```
}

public void gameOver() {
    end = true;
    exit = true;
    clickable = false;

    submitMsg1L.setVisible(false);
    submitMsg2L.setVisible(false);

    setBoxNormal(1, 1, 1, 1);
    switch (sequence[currentLevel]) {
        case 1:
            setBoxHighlight(1, 0, 0, 0);
            break;
        case 2:
            setBoxHighlight(0, 1, 0, 0);
            break;
        case 3:
            setBoxHighlight(0, 0, 1, 0);
            break;
        case 4:
            setBoxHighlight(0, 0, 0, 1);
            break;
    }

    playAudio(7);
    JOptionPane.showMessageDialog(this, "Wrong !!");

    setBoxNormal(1, 1, 1, 1);
    clearClickHistory();

    gameScoreL.setText("\nGame Score: " + gameScore);
    highScoreL.setText("\nHigh Score: " + highScore);

    gamePanel.setVisible(false);
    gameOverPanel.setVisible(true);
}

public void clearClickHistory() {
    click15TF.setBackground(Color.decode("0XF0F0F0"));
    click14TF.setBackground(Color.decode("0XF0F0F0"));
    click13TF.setBackground(Color.decode("0XF0F0F0"));
    click12TF.setBackground(Color.decode("0XF0F0F0"));
    click11TF.setBackground(Color.decode("0XF0F0F0"));
    click10TF.setBackground(Color.decode("0XF0F0F0"));
    click09TF.setBackground(Color.decode("0XF0F0F0"));
    click08TF.setBackground(Color.decode("0XF0F0F0"));
    click07TF.setBackground(Color.decode("0XF0F0F0"));
    click06TF.setBackground(Color.decode("0XF0F0F0"));
    click05TF.setBackground(Color.decode("0XF0F0F0"));
    click04TF.setBackground(Color.decode("0XF0F0F0"));
    click03TF.setBackground(Color.decode("0XF0F0F0"));
    click02TF.setBackground(Color.decode("0XF0F0F0"));
    click01TF.setBackground(Color.decode("0XF0F0F0"));
}
```

```
}  
  
public void updateClickHistory(int c) {  
    click15TF.setBackground(click14TF.getBackground());  
    click14TF.setBackground(click13TF.getBackground());  
    click13TF.setBackground(click12TF.getBackground());  
    click12TF.setBackground(click11TF.getBackground());  
    click11TF.setBackground(click10TF.getBackground());  
    click10TF.setBackground(click09TF.getBackground());  
    click09TF.setBackground(click08TF.getBackground());  
    click08TF.setBackground(click07TF.getBackground());  
    click07TF.setBackground(click06TF.getBackground());  
    click06TF.setBackground(click05TF.getBackground());  
    click05TF.setBackground(click04TF.getBackground());  
    click04TF.setBackground(click03TF.getBackground());  
    click03TF.setBackground(click02TF.getBackground());  
    click02TF.setBackground(click01TF.getBackground());  
    if (c == 1) {  
        click01TF.setBackground(Color.decode("0xFF0000"));  
    }  
    if (c == 2) {  
        click01TF.setBackground(Color.decode("0x00FF00"));  
    }  
    if (c == 3) {  
        click01TF.setBackground(Color.decode("0x0000FF"));  
    }  
    if (c == 4) {  
        click01TF.setBackground(Color.decode("0xFFFF00"));  
    }  
}  
  
public void setBoxNormal(int b1, int b2, int b3, int b4) {  
    if (b1 == 1) {  
        box1TF.setBackground(Color.decode("0x770000"));  
    }  
    if (b2 == 1) {  
        box2TF.setBackground(Color.decode("0x007700"));  
    }  
    if (b3 == 1) {  
        box3TF.setBackground(Color.decode("0x000077"));  
    }  
    if (b4 == 1) {  
        box4TF.setBackground(Color.decode("0X777700"));  
    }  
}  
  
public void setBoxPressed(int b1, int b2, int b3, int b4) {  
    if (b1 == 1) {  
        box1TF.setBackground(Color.decode("0xCC0000"));  
    }  
    if (b2 == 1) {  
        box2TF.setBackground(Color.decode("0x00CC00"));  
    }  
    if (b3 == 1) {  
        box3TF.setBackground(Color.decode("0x0000CC"));  
    }  
}
```

```
    }
    if (b4 == 1) {
        box4TF.setBackground(Color.decode("0XCCCC00"));
    }
}

public void setBoxHighlight(int b1, int b2, int b3, int b4) {
    if (b1 == 1) {
        box1TF.setBackground(Color.decode("0xFF0000"));
    }
    if (b2 == 1) {
        box2TF.setBackground(Color.decode("0x00FF00"));
    }
    if (b3 == 1) {
        box3TF.setBackground(Color.decode("0x0000FF"));
    }
    if (b4 == 1) {
        box4TF.setBackground(Color.decode("0xFFFF00"));
    }
}

public void dataPass1(String uName, int sc) {
    username = uName;
    highScore = sc;
    nameL.setText(username);
    highScoreL2.setText("" + highScore);
}

public void playAudio(int track) {
    switch (track) {
        case 0:
            Toolkit.getDefaultToolkit().beep();
            break;
        case 1:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\proceedToGame.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 2:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\singleBeep.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 3:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\glassPing.wav"));
                AudioStream audio = new AudioStream(music);
            }
    }
}
```



```

        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
case 4:
    try {
        InputStream music = new FileInputStream(new File("src\\Audio\\singleCoinDrop.wav"));
        AudioStream audio = new AudioStream(music);
        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
case 5:
    try {
        InputStream music = new FileInputStream(new File("src\\Audio\\multipleCoinDrop.wav"));
        AudioStream audio = new AudioStream(music);
        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
case 6:
    try {
        InputStream music = new FileInputStream(new File("src\\Audio\\tadaHighScore.wav"));
        AudioStream audio = new AudioStream(music);
        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
case 7:
    try {
        InputStream music = new FileInputStream(new File("src\\Audio\\buzzerGameOver.wav"));
        AudioStream audio = new AudioStream(music);
        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
    }
}

```

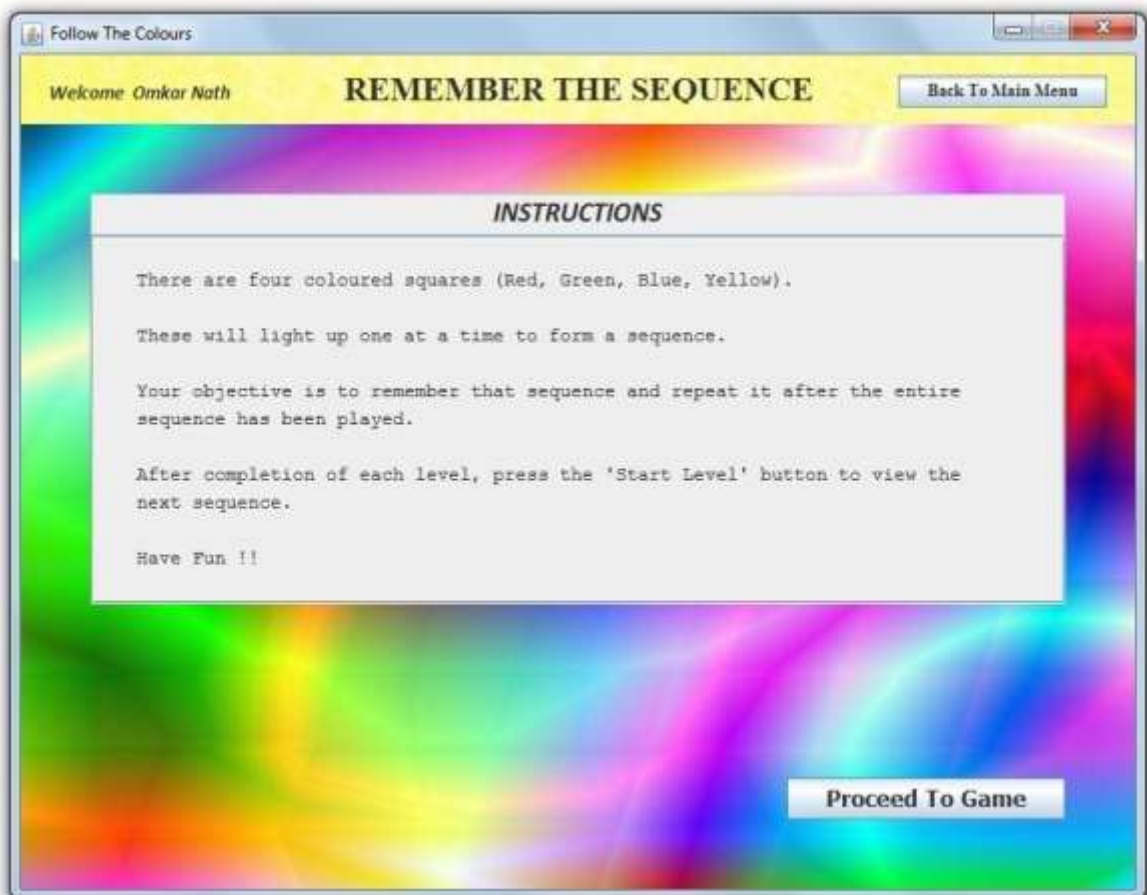
## Instructions:

### Button – Back To Main Menu:

```

private void returnBtnActionPerformed(java.awt.event.ActionEvent evt) {
    if (!exit) {
        playAudio(0);
        int reply = JOptionPane.showConfirmDialog(this, "Game In Progress !!\nAre you sure you want to
exit", "Caution !!", JOptionPane.YES_NO_OPTION);
        if (reply == JOptionPane.YES_OPTION) {

```



Instructions of the Game

```

        exit = true;
    }
}

if (exit) {
    homePage frame0 = new homePage();
    rememberTheSequence.this.setVisible(false);
    frame0.setVisible(true);
    frame0.dataPass0(username);
}
}

```

### Button – Proceed To Game:

```

private void proceedToGameBtnActionPerformed(java.awt.event.ActionEvent evt) {
    instructionsPanel.setVisible(false);

    gamePanel.setVisible(true);

    playAudio(1);
}

```

### Playing The Game:

#### Button – Start Game / Start Level:

```

private void startBtnActionPerformed(java.awt.event.ActionEvent evt) {
    startBtn.setVisible(false);

    startMsg1L.setVisible(false);
    startMsg2L.setVisible(false);

    exit = false;

    sequence = new int[gameLevel + 1];

    for (int i = 1; i <= gameLevel; i++) {
        sequence[i] = (int) (Math.random() * 4 + 1);
    }

    playSequence(msTimeBegin, msTimeBegin + msTimeFlash);
}

```

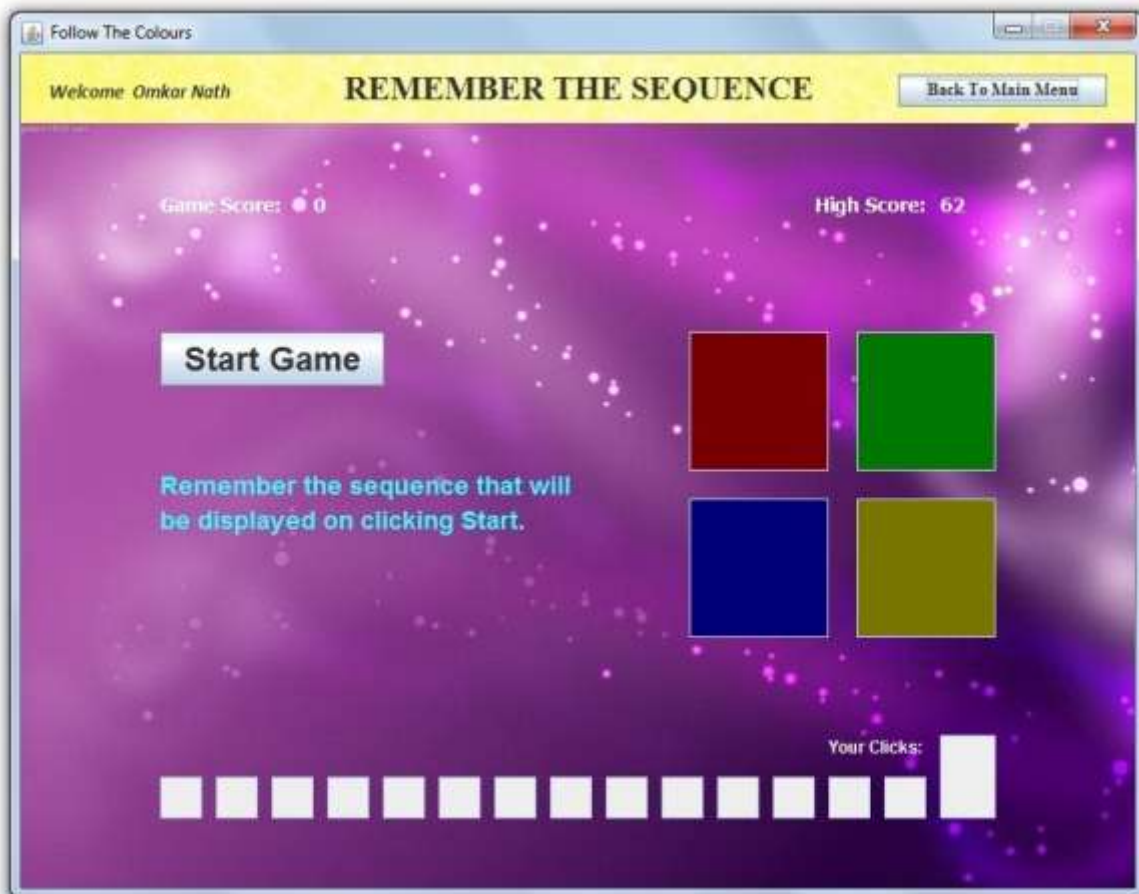
#### Text Field – 1 (Red):

```

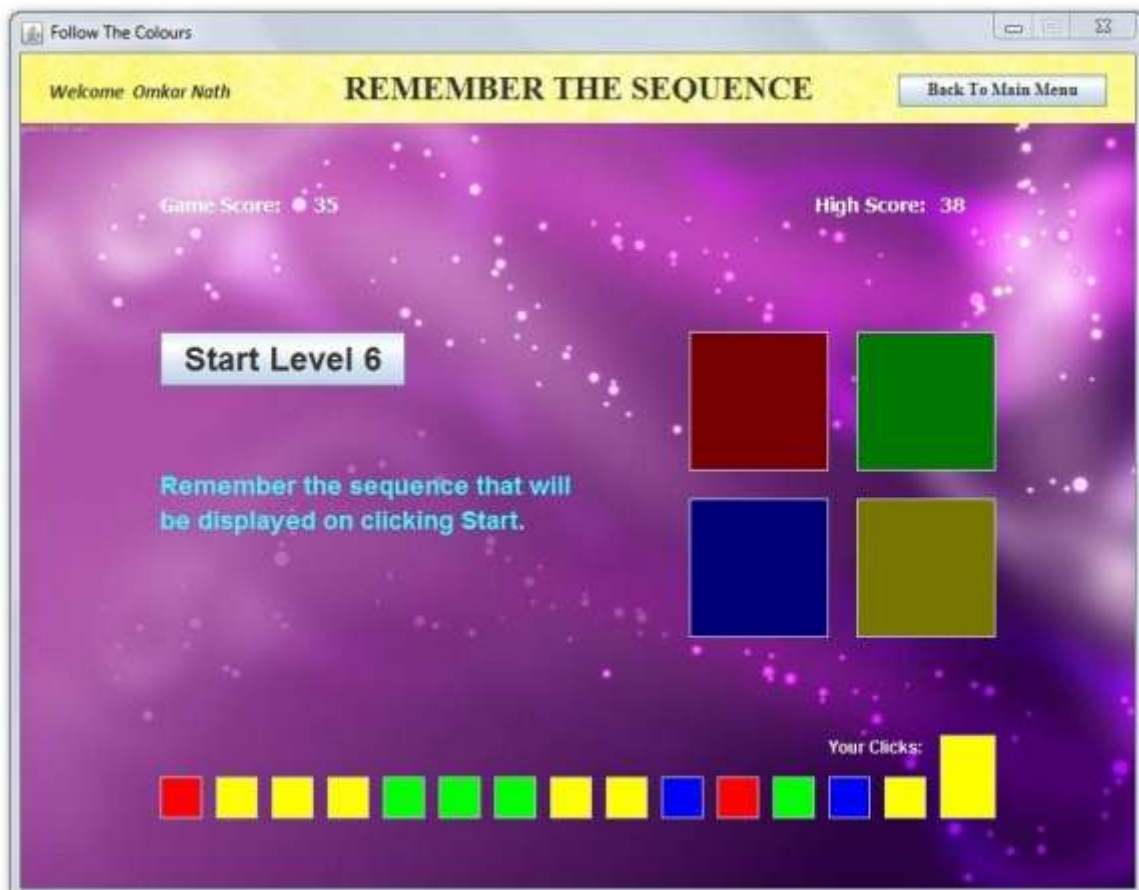
private void box1TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlight(1, 0, 0, 0);
    }
}

private void box1TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable) {

```



Starting the Game



Starting a New Level

```

        setBoxNormal(1, 0, 0, 0);
    }
}

private void box1TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(1, 0, 0, 0);
    }
}

private void box1TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlight(1, 0, 0, 0);
    }
}

private void box1TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        updateClickHistory(1);

        if (sequence[currentLevel] == 1) {
            playAudio(4);

            gameScore += currentLevel;
            gameScoreL2.setText(Integer.toString(gameScore));

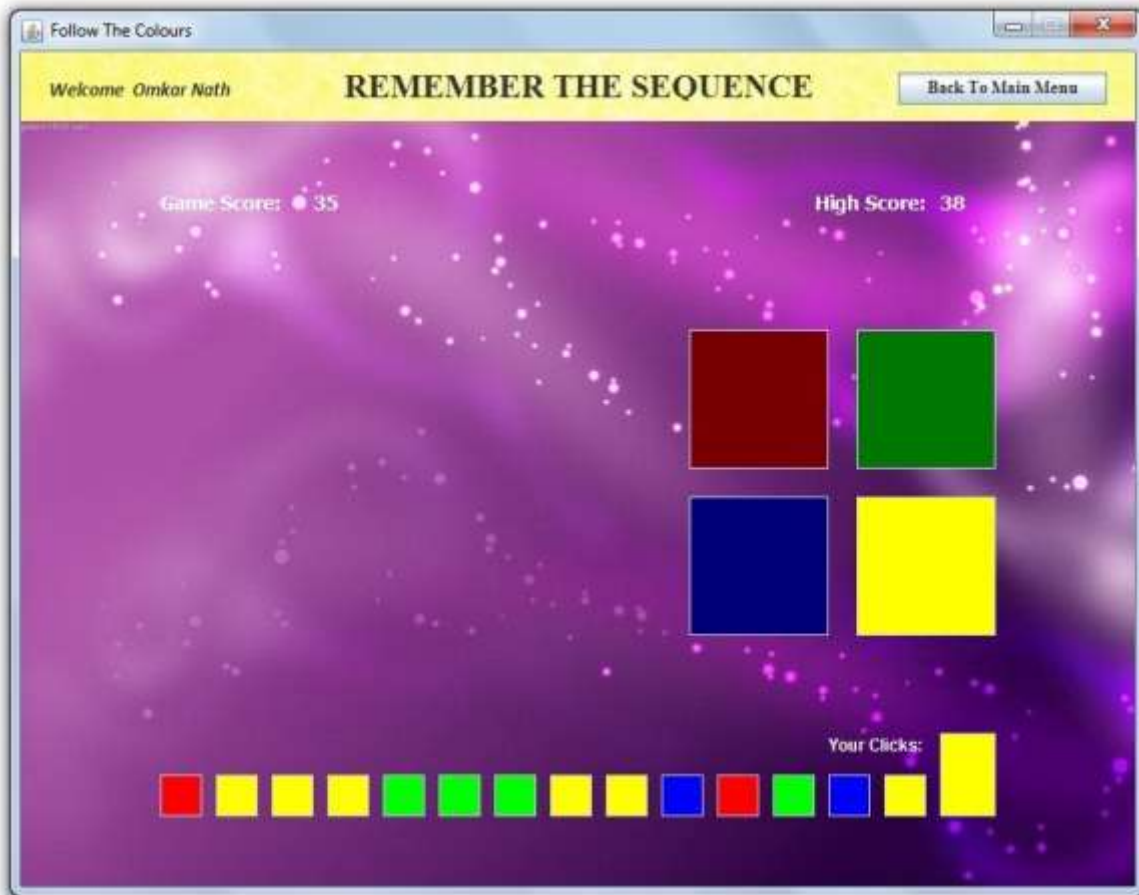
            if (gameScore > highScore) {

                if (!newHighScore && highScore != 0) {
                    newHighScore = true;
                    playAudio(6);
                    JOptionPane.showMessageDialog(this, "New High Score achieved !!");
                }

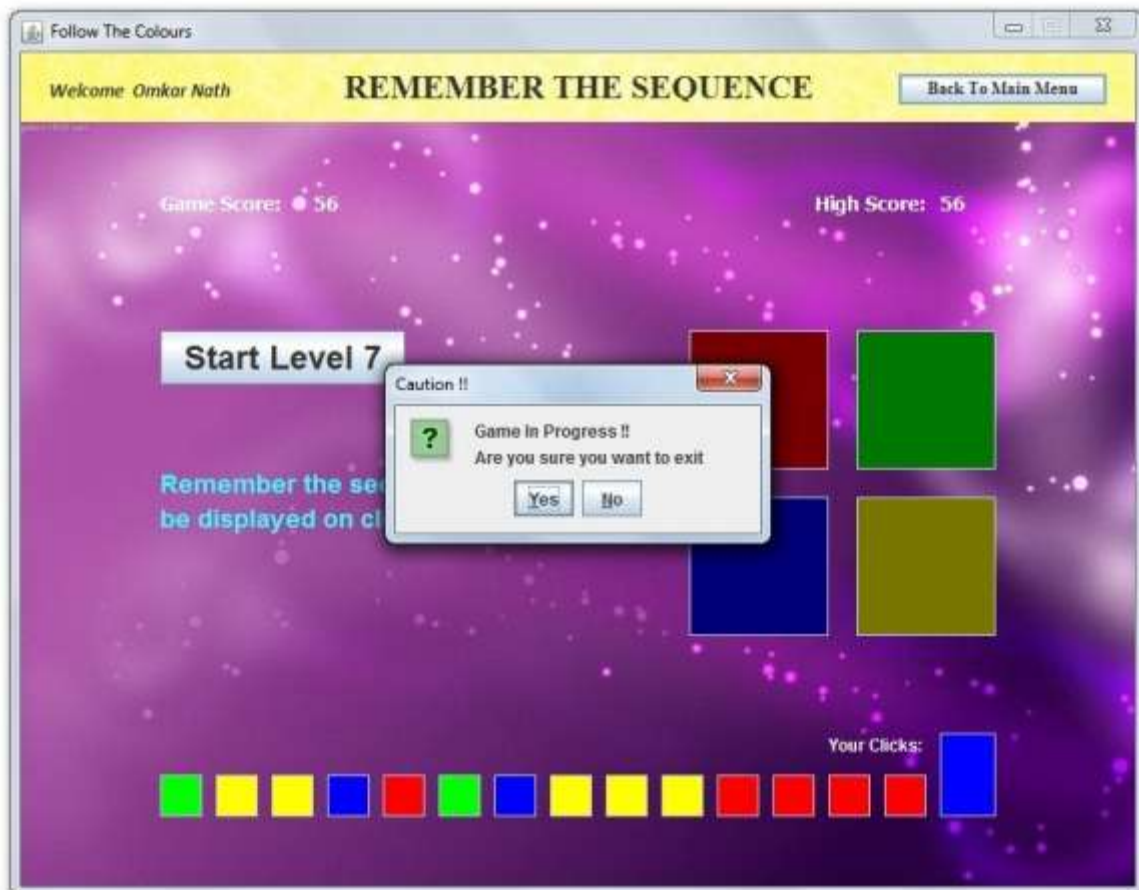
                highScore = gameScore;
                highScoreL2.setText(Integer.toString(highScore));

                try {
                    Class.forName("java.sql.DriverManager");
                    Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
                    Statement stmt = (Statement) con.createStatement();
                    String query = "UPDATE highscores SET RememberTheSequence = " + highScore + "
WHERE Username = " + username + ",";
                    stmt.executeUpdate(query);
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(this, e.getMessage());
                }
            }
            if (++currentLevel > gameLevel) {
                newLevel();
            }
        } else {
            gameOver();
        }
    }
}

```



Sequence Being Shown



Message when Player wants to quit while Playing

```

    }
}

```

### Text Field – 2 (Green):

```

private void box2TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlight(0, 1, 0, 0);
    }
}

private void box2TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxNormal(0, 1, 0, 0);
    }
}

private void box2TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(0, 1, 0, 0);
    }
}

private void box2TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlight(0, 1, 0, 0);
    }
}

private void box2TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        updateClickHistory(2);

        if (sequence[currentLevel] == 2) {
            playAudio(4);

            gameScore += currentLevel;
            gameScoreL2.setText(Integer.toString(gameScore));

            if (gameScore > highScore) {

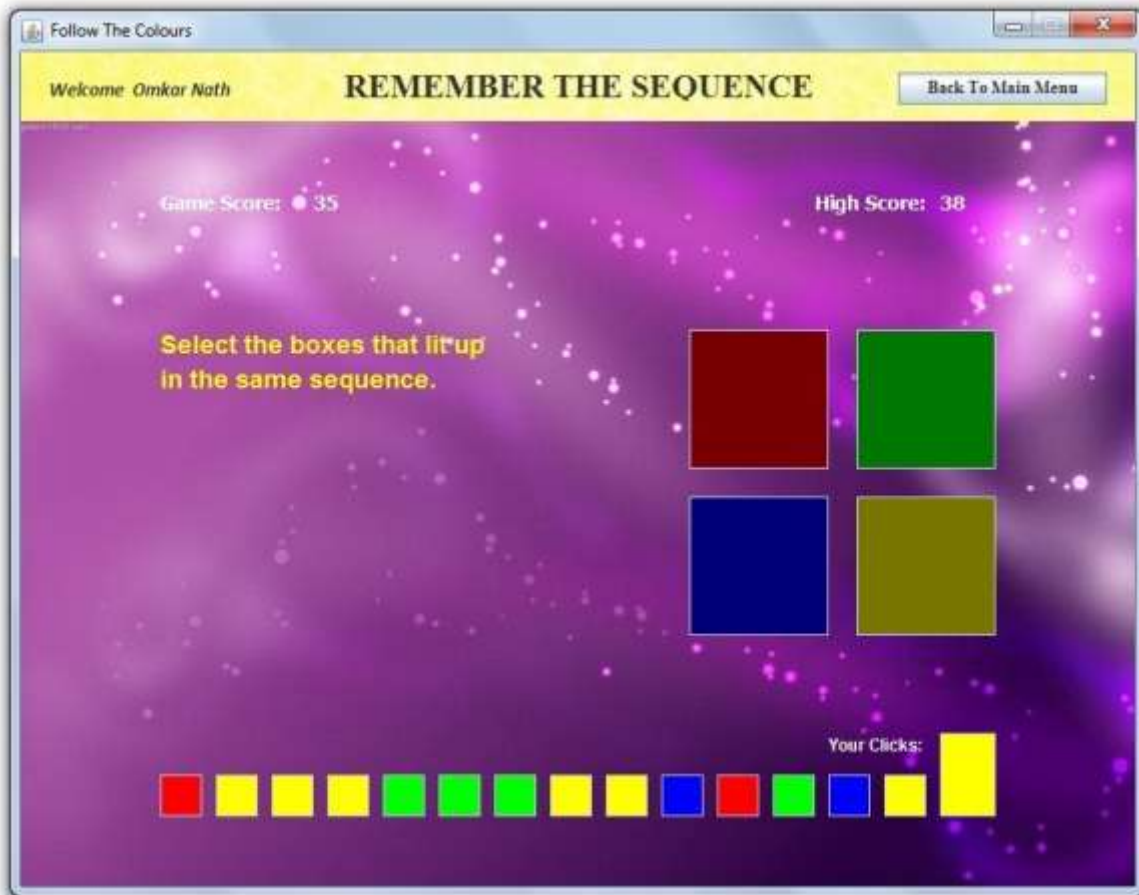
                if (!newHighScore && highScore != 0) {
                    newHighScore = true;
                    playAudio(6);
                    JOptionPane.showMessageDialog(this, "New High Score achieved !!");
                }

                highScore = gameScore;
                highScoreL2.setText(Integer.toString(highScore));

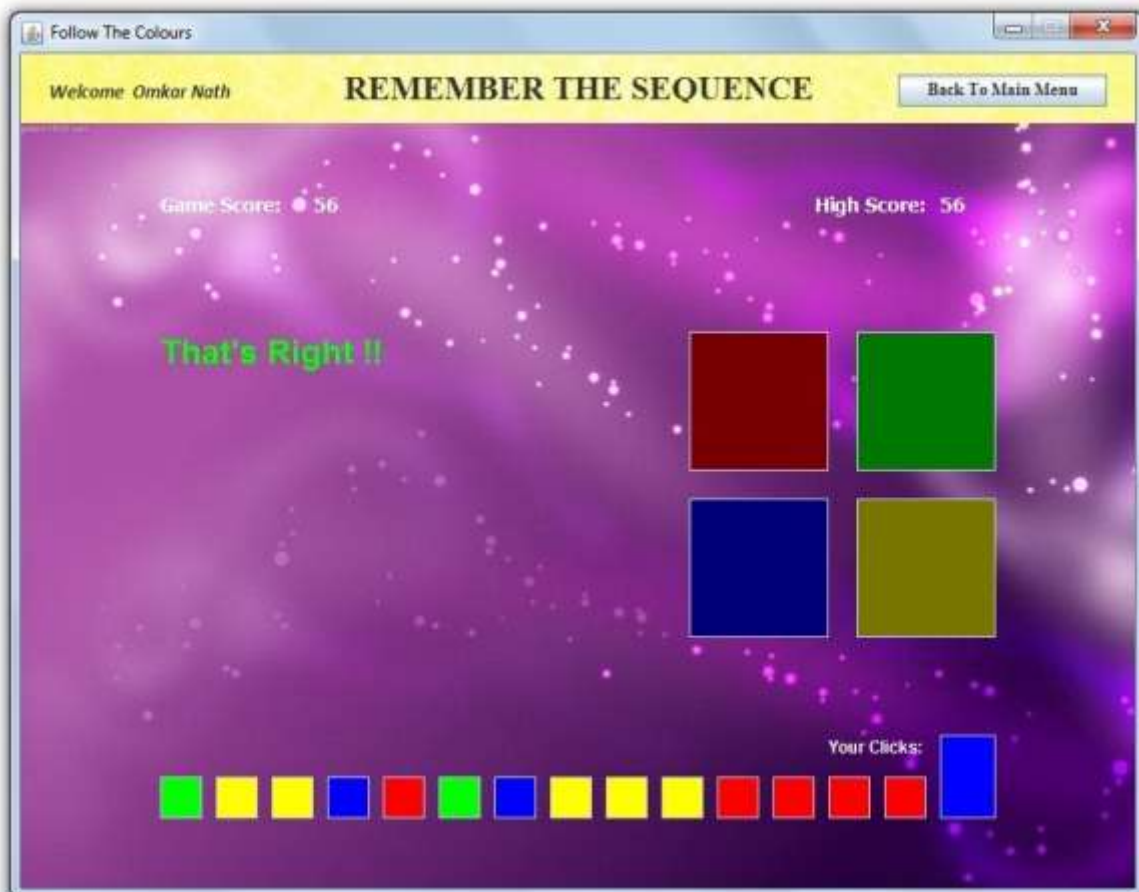
                try {
                    Class.forName("java.sql.DriverManager");
                    Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");

```





Waiting for User to Click Boxes as per Sequence



Message when Player Gets the Right Sequence



```

        Statement stmt = (Statement) con.createStatement();
        String query = "UPDATE highscores SET RememberTheSequence = " + highScore + "
WHERE Username = " + username + " ";
        stmt.executeUpdate(query);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
}
if (++currentLevel > gameLevel) {
    newLevel();
}
} else {
    gameOver();
}
}
}

```

### Text Field – 3 (Blue):

```

private void box3TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlight(0, 0, 1, 0);
    }
}

private void box3TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxNormal(0, 0, 1, 0);
    }
}

private void box3TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(0, 0, 1, 0);
    }
}

private void box3TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlight(0, 0, 1, 0);
    }
}

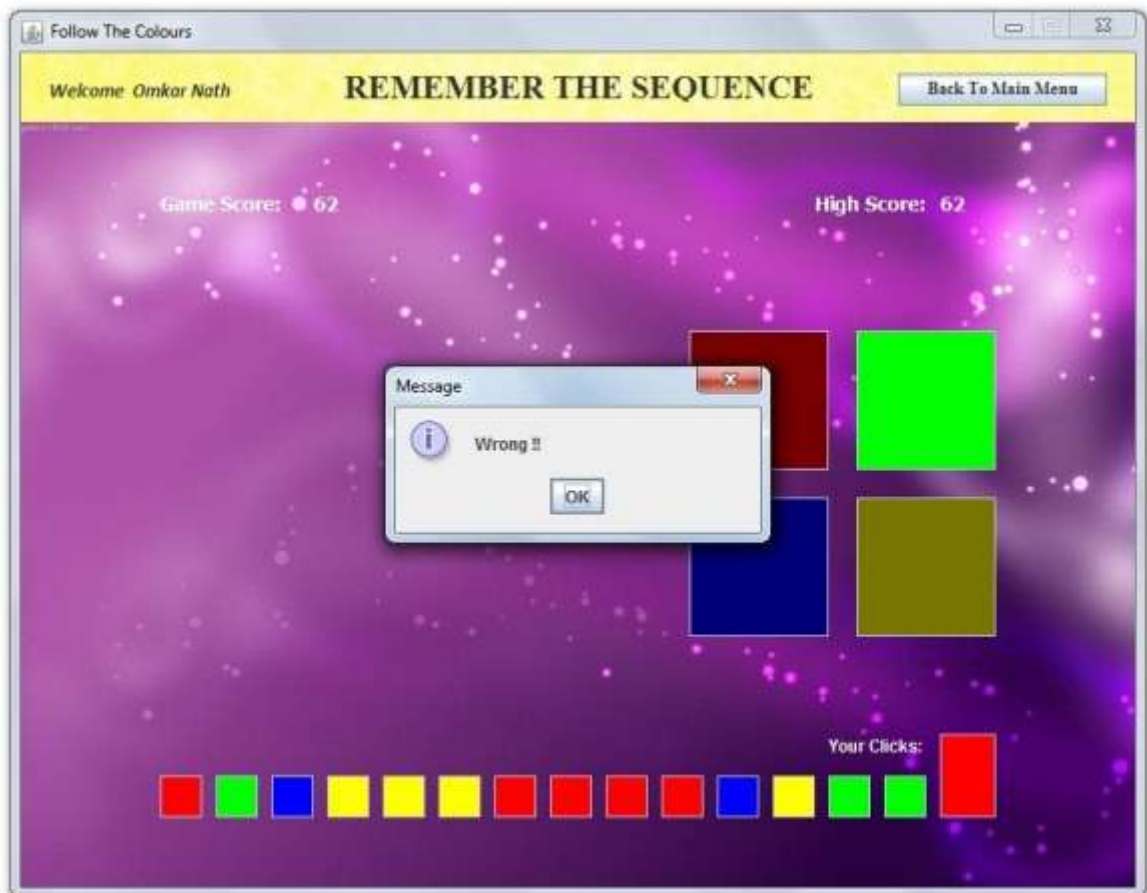
private void box3TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        updateClickHistory(3);

        if (sequence[currentLevel] == 3) {
            playAudio(4);

            gameScore += currentLevel;
            gameScoreL2.setText(Integer.toString(gameScore));

            if (gameScore > highScore) {

```



Message when the Player Enters the Wrong Sequence

```

        if (!newHighScore && highScore != 0) {
            newHighScore = true;
            playAudio(6);
            JOptionPane.showMessageDialog(this, "New High Score achieved !!");
        }

        highScore = gameScore;
        highScoreL2.setText(Integer.toString(highScore));

        try {
            Class.forName("java.sql.DriverManager");
            Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
            Statement stmt = (Statement) con.createStatement();
            String query = "UPDATE highscores SET RememberTheSequence = " + highScore + "
WHERE Username = " + username + " ";
            stmt.executeUpdate(query);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
    }
    if (++currentLevel > gameLevel) {
        newLevel();
    }
    else {
        gameOver();
    }
}
}
}

```

#### Text Field – 4 (Yellow):

```

private void box4TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlight(0, 0, 0, 1);
    }
}

private void box4TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxNormal(0, 0, 0, 1);
    }
}

private void box4TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(0, 0, 0, 1);
    }
}

private void box4TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlight(0, 0, 0, 1);
    }
}

```

```

    }
}

private void box4TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        updateClickHistory(4);
        if (sequence[currentLevel] == 4) {
            playAudio(4);

            gameScore += currentLevel;
            gameScoreL2.setText(Integer.toString(gameScore));

            if (gameScore > highScore) {

                if (!newHighScore && highScore != 0) {
                    newHighScore = true;
                    playAudio(6);
                    JOptionPane.showMessageDialog(this, "New High Score achieved !!");
                }

                highScore = gameScore;
                highScoreL2.setText(Integer.toString(highScore));

                try {
                    Class.forName("java.sql.DriverManager");
                    Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
                    Statement stmt = (Statement) con.createStatement();
                    String query = "UPDATE highscores SET RememberTheSequence = " + highScore + "
WHERE Username = " + username + " ";
                    stmt.executeUpdate(query);
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(this, e.getMessage());
                }
            }
            if (++currentLevel > gameLevel) {
                newLevel();
            }
        } else {
            gameOver();
        }
    }
}

```

## Game Over:

### Button – Play Again:

```

private void playAgainBtnActionPerformed(java.awt.event.ActionEvent evt) {
    gameOverPanel.setVisible(false);

    clickable = false;
    exit = true;
}

```



End of Game

```
gameLevel = 1;

gameScore = 0;
gameScoreL2.setText(Integer.toString(gameScore));

highScoreL2.setText(Integer.toString(highScore));

if (highScore == 0) {
    newHighScore = true;
} else {
    newHighScore = false;
}

correctMsgL.setVisible(false);
submitMsg1L.setVisible(false);
submitMsg2L.setVisible(false);

startBtn.setText("Start Game");
startBtn.setVisible(true);

startMsg1L.setVisible(true);
startMsg2L.setVisible(true);

gamePanel.setVisible(true);

playAudio(1);
}
```

## **Frame: Get the Squares (Game 2)**

### **Importing Java Packages:**

```
import com.mysql.jdbc.Connection;  
import com.mysql.jdbc.Statement;  
import java.sql.DriverManager;
```

```
import java.awt.*;
```

```
import java.io.File;  
import java.io.FileInputStream;  
import java.io.InputStream;  
import sun.audio.AudioPlayer;  
import sun.audio.AudioStream;
```

```
import java.util.*;
```

```
import javax.swing.JOptionPane;
```

### **Global Variables:**

```
Timer timerDelay;  
Timer timerFlash;  
Timer timerCorrect;
```

```
int msTimeBegin = 750;  
int msTimeEnd = 1500;  
int msTimeMaximum = 2200;  
int msTimeMinimum = 200;  
int msTimeStep = 200;  
int msTimeInterval;  
int maxMultiple = 2;
```

```
String colourNormal;  
String colourHighlighted;  
String colourPressed;  
String username = "";
```

```
boolean begin = true;  
boolean clickable;  
boolean newHighScore;  
boolean exit;  
boolean selected[] = new boolean[17];  
boolean generated[] = new boolean[17];
```

```
int gameScore;  
int highScore;  
int numberOfBoxes;
```

## Event – Form Window Activated:

```
private void formWindowActivated(java.awt.event.WindowEvent evt) {  
    if (begin) {  
        begin = false;  
        exit = true;  
  
        msTimeInterval = msTimeMaximum;  
  
        colourNormal = "0x770000";  
        colourHighlighted = "0xFF0000";  
        colourPressed = "0xCC0000";  
  
        gameScore = 0;  
        gameScore2L.setText(Integer.toString(gameScore));  
  
        correctMsgL.setVisible(false);  
  
        submitBtn.setVisible(false);  
        submitMsg1L.setVisible(false);  
        submitMsg2L.setVisible(false);  
  
        startMsg1L.setVisible(true);  
        startMsg2L.setVisible(true);  
  
        gamePanel.setVisible(false);  
        gameOverPanel.setVisible(false);  
  
        coverPanel.setVisible(false);  
  
        clickable = false;  
        if (highScore == 0) {  
            newHighScore = true;  
        } else {  
            newHighScore = false;  
        }  
  
        for (int i = 1; i <= 16; i++) {  
            setBoxNormal(i);  
        }  
    }  
}
```

## Methods & Functions Used:

```
public void showBoxes(int msDelay, int msFlash) {  
    timerDelay = new Timer();  
    timerFlash = new Timer();  
    timerDelay.schedule(new taskDelay(), msDelay);  
    timerFlash.schedule(new taskFlash(), msFlash);  
}  
  
class taskDelay extends TimerTask {  
    public void run() {
```



```
        playAudio(3);
        for (int i = 1; i <= 16; i++) {
            if (generated[i]) {
                setBoxHighlighted(i);
            }
        }
    }
}

class taskFlash extends TimerTask {
    public void run() {
        for (int i = 1; i <= 16; i++) {
            setBoxNormal(i);
        }
        clickable = true;
        submitBtn.setVisible(true);
        submitMsg1L.setVisible(true);
        submitMsg2L.setVisible(true);
    }
}

public void nextLevel(int msTime) {
    timerCorrect = new Timer();
    timerCorrect.schedule(new taskCorrect(), msTime);
}

class taskCorrect extends TimerTask {
    public void run() {
        int i;
        for (i = 1; i <= 16; i++) {
            setBoxNormal(i);
        }

        correctMsgL.setVisible(false);

        startBtn.setText("Start");
        startBtn.setVisible(true);

        startMsg1L.setVisible(true);
        startMsg2L.setVisible(true);
    }
}

public void gameOver() {
    gameScoreL.setText("\nGame Score: " + gameScore);
    highScoreL.setText("\nHigh Score: " + highScore);

    gamePanel.setVisible(false);
    gameOverPanel.setVisible(true);
}

public void setBoxNormal(int box) {
    if (box == 1) {
        box01TF.setBackground(Color.decode(colourNormal));
    } else if (box == 2) {
```

```
        box02TF.setBackground(Color.decode(colourNormal));
    } else if (box == 3) {
        box03TF.setBackground(Color.decode(colourNormal));
    } else if (box == 4) {
        box04TF.setBackground(Color.decode(colourNormal));
    } else if (box == 5) {
        box05TF.setBackground(Color.decode(colourNormal));
    } else if (box == 6) {
        box06TF.setBackground(Color.decode(colourNormal));
    } else if (box == 7) {
        box07TF.setBackground(Color.decode(colourNormal));
    } else if (box == 8) {
        box08TF.setBackground(Color.decode(colourNormal));
    } else if (box == 9) {
        box09TF.setBackground(Color.decode(colourNormal));
    } else if (box == 10) {
        box10TF.setBackground(Color.decode(colourNormal));
    } else if (box == 11) {
        box11TF.setBackground(Color.decode(colourNormal));
    } else if (box == 12) {
        box12TF.setBackground(Color.decode(colourNormal));
    } else if (box == 13) {
        box13TF.setBackground(Color.decode(colourNormal));
    } else if (box == 14) {
        box14TF.setBackground(Color.decode(colourNormal));
    } else if (box == 15) {
        box15TF.setBackground(Color.decode(colourNormal));
    } else if (box == 16) {
        box16TF.setBackground(Color.decode(colourNormal));
    }
}
```

```
public void setBoxHighlighted(int box) {
    if (box == 1) {
        box01TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 2) {
        box02TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 3) {
        box03TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 4) {
        box04TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 5) {
        box05TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 6) {
        box06TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 7) {
        box07TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 8) {
        box08TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 9) {
        box09TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 10) {
        box10TF.setBackground(Color.decode(colourHighlighted));
    } else if (box == 11) {
        box11TF.setBackground(Color.decode(colourHighlighted));
    }
}
```

```
    } else if (box == 12) {  
        box12TF.setBackground(Color.decode(colourHighlighted));  
    } else if (box == 13) {  
        box13TF.setBackground(Color.decode(colourHighlighted));  
    } else if (box == 14) {  
        box14TF.setBackground(Color.decode(colourHighlighted));  
    } else if (box == 15) {  
        box15TF.setBackground(Color.decode(colourHighlighted));  
    } else if (box == 16) {  
        box16TF.setBackground(Color.decode(colourHighlighted));  
    }  
}
```

**public void setBoxPressed(int box) {**

```
    if (box == 1) {  
        box01TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 2) {  
        box02TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 3) {  
        box03TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 4) {  
        box04TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 5) {  
        box05TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 6) {  
        box06TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 7) {  
        box07TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 8) {  
        box08TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 9) {  
        box09TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 10) {  
        box10TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 11) {  
        box11TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 12) {  
        box12TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 13) {  
        box13TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 14) {  
        box14TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 15) {  
        box15TF.setBackground(Color.decode(colourPressed));  
    } else if (box == 16) {  
        box16TF.setBackground(Color.decode(colourPressed));  
    }  
}
```

**public void dataPass2(String uName, int sc) {**

```
    username = uName;  
    highScore = sc;  
    nameL.setText(username);  
    highScore2L.setText("" + highScore);  
}
```

```
public void playAudio(int track) {
    switch (track) {
        case 0:
            Toolkit.getDefaultToolkit().beep();
            break;
        case 1:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\proceedToGame.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 2:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\singleBeep.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 3:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\glassPing.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 4:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\singleCoinDrop.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 5:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\multipleCoinDrop.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 6:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\tadaHighScore.wav"));
                AudioStream audio = new AudioStream(music);
            }
    }
}
```

```

        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
case 7:
    try {
        InputStream music = new FileInputStream(new File("src\\Audio\\buzzerGameOver.wav"));
        AudioStream audio = new AudioStream(music);
        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
}
}
}

```

## Instructions:

### Button – Back To Main Menu:

```

private void returnBtnActionPerformed(java.awt.event.ActionEvent evt) {
    if (!exit) {
        playAudio(0);
        int reply = JOptionPane.showConfirmDialog(this, "Game In Progress !!\nAre you sure you want to
exit", "Caution !!", JOptionPane.YES_NO_OPTION);
        if (reply == JOptionPane.YES_OPTION) {
            exit = true;
        }
    }

    if (exit) {
        homePage frame0 = new homePage();
        getTheSquares.this.setVisible(false);
        frame0.setVisible(true);
        frame0.dataPass0(username);
    }
}

```

### Button – Proceed To Game:

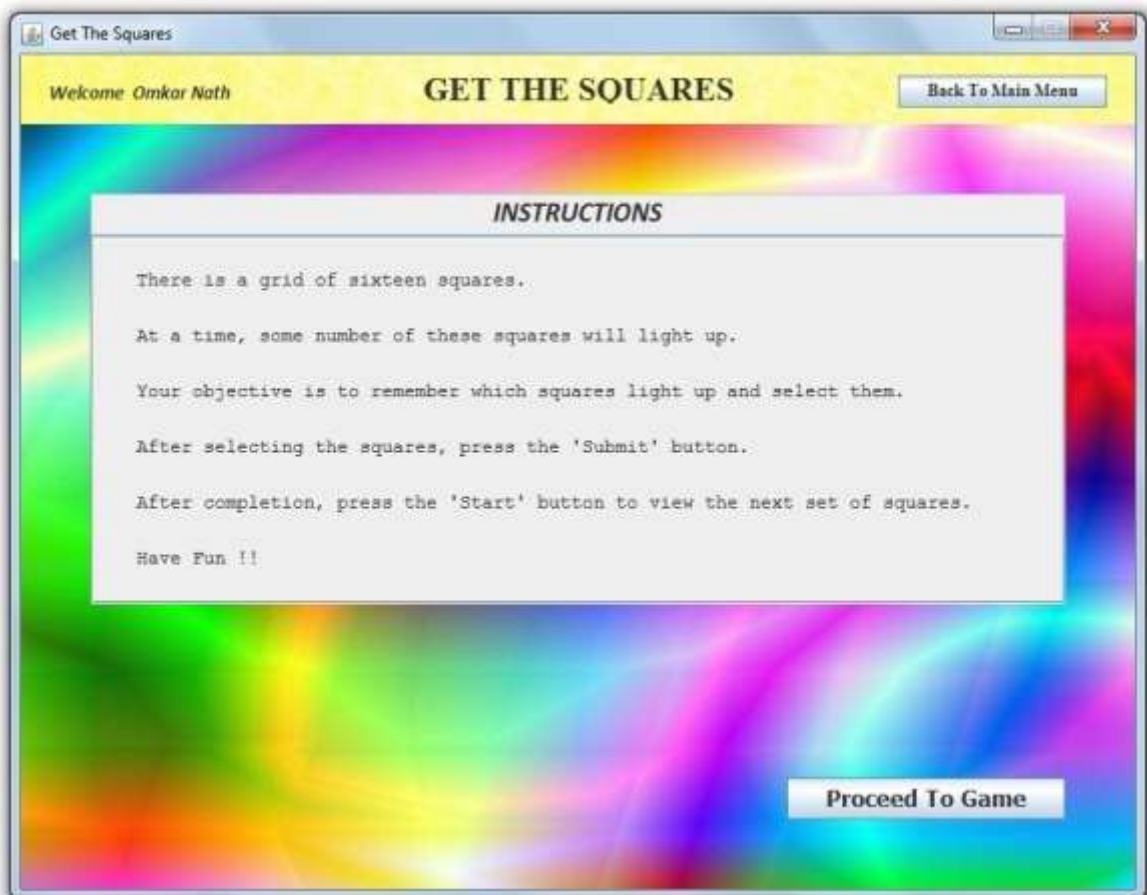
```

private void proceedToGameActionPerformed(java.awt.event.ActionEvent evt) {
    instructionsPanel.setVisible(false);

    gamePanel.setVisible(true);

    playAudio(1);
}

```



Instructions of the Game

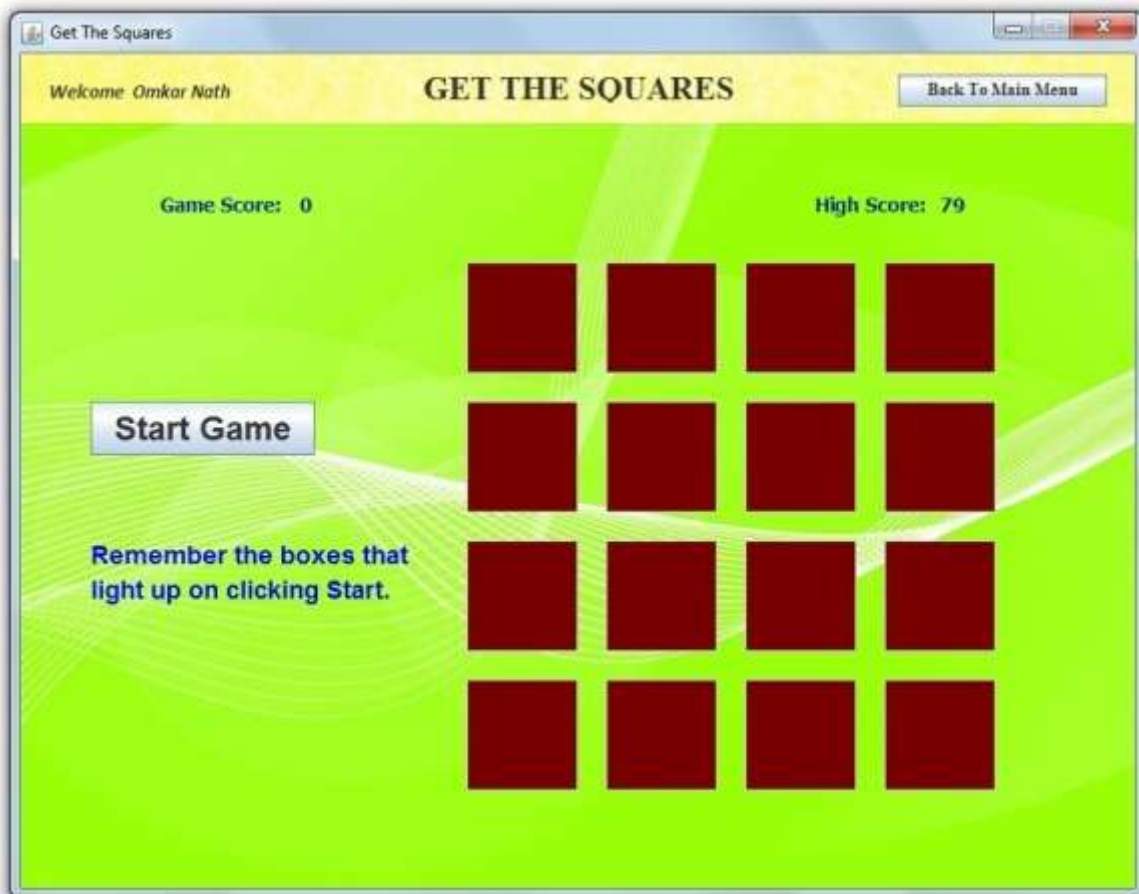
## Game:

### Button – Start Game/ Start:

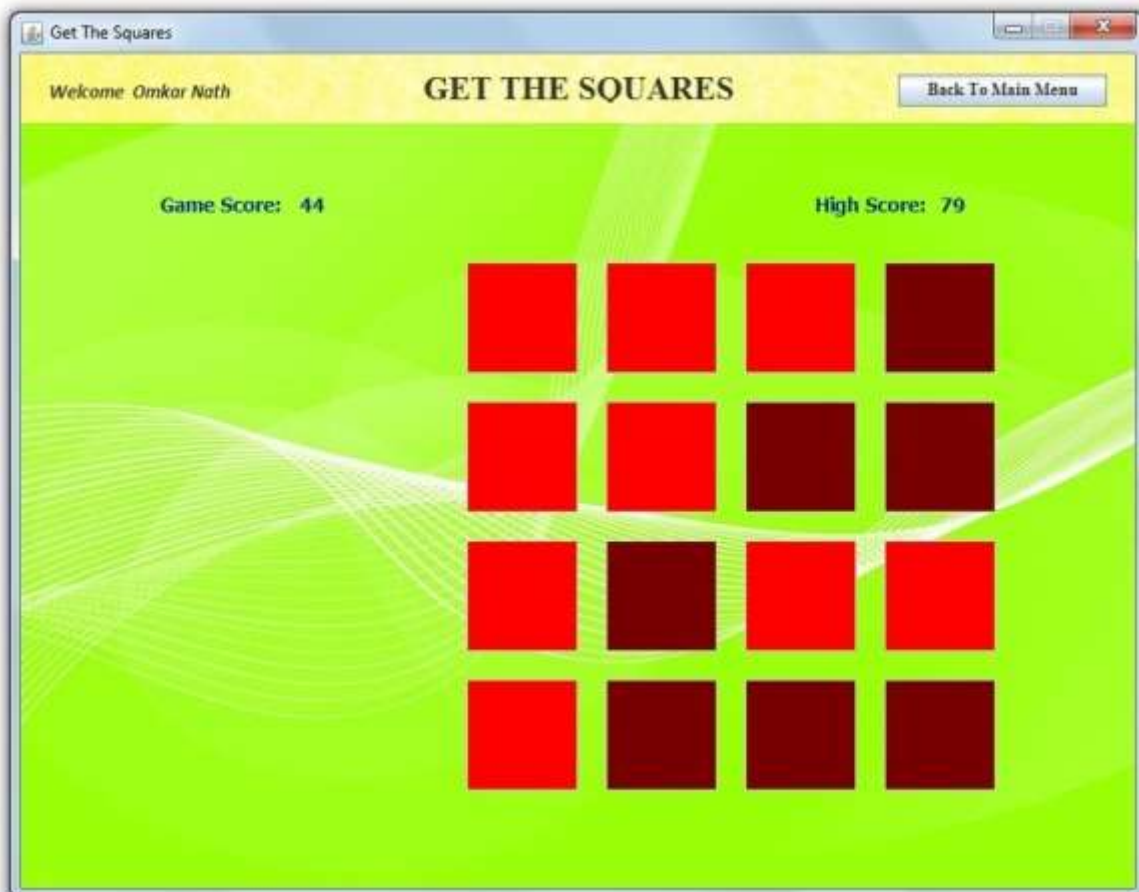
```
private void startBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    int i;  
    int box;  
  
    exit = false;  
  
    startBtn.setVisible(false);  
  
    startMsg1L.setVisible(false);  
    startMsg2L.setVisible(false);  
  
    for (i = 1; i <= 16; i++) {  
        selected[i] = false;  
        generated[i] = false;  
    }  
  
    numberOfBoxes = (int) (Math.random() * 16) + 1;  
  
    i = numberOfBoxes;  
    while (i > 0) {  
        box = (int) (Math.random() * 16) + 1;  
        if (!generated[box]) {  
            generated[box] = true;  
            i--;  
        }  
    }  
  
    showBoxes(msTimeBegin, msTimeBegin + msTimeInterval);  
}
```

### Button – Submit:

```
private void submitBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    int i;  
    int temp;  
    boolean flag = false;  
  
    for (i = 1; i <= 16; i++) {  
        if (generated[i] != selected[i]) {  
            flag = true;  
            break;  
        }  
    }  
  
    if (flag) {  
        clickable = false;  
  
        for (i = 1; i <= 16; i++) {  
            setBoxNormal(i);  
        }  
    }  
}
```



Starting the Game



Auto Generated Random Pattern



```
    }
    for (i = 1; i <= 16; i++) {
        if (generated[i]) {
            setBoxHighlighted(i);
        }
    }

    submitBtn.setVisible(false);
    submitMsg1L.setVisible(false);
    submitMsg2L.setVisible(false);

    playAudio(7);
    JOptionPane.showMessageDialog(this, "Wrong !!");

    for (i = 1; i <= 16; i++) {
        setBoxNormal(i);
    }

    exit = true;

    gameOver();
} else {
    clickable = false;

    submitBtn.setVisible(false);
    submitMsg1L.setVisible(false);
    submitMsg2L.setVisible(false);

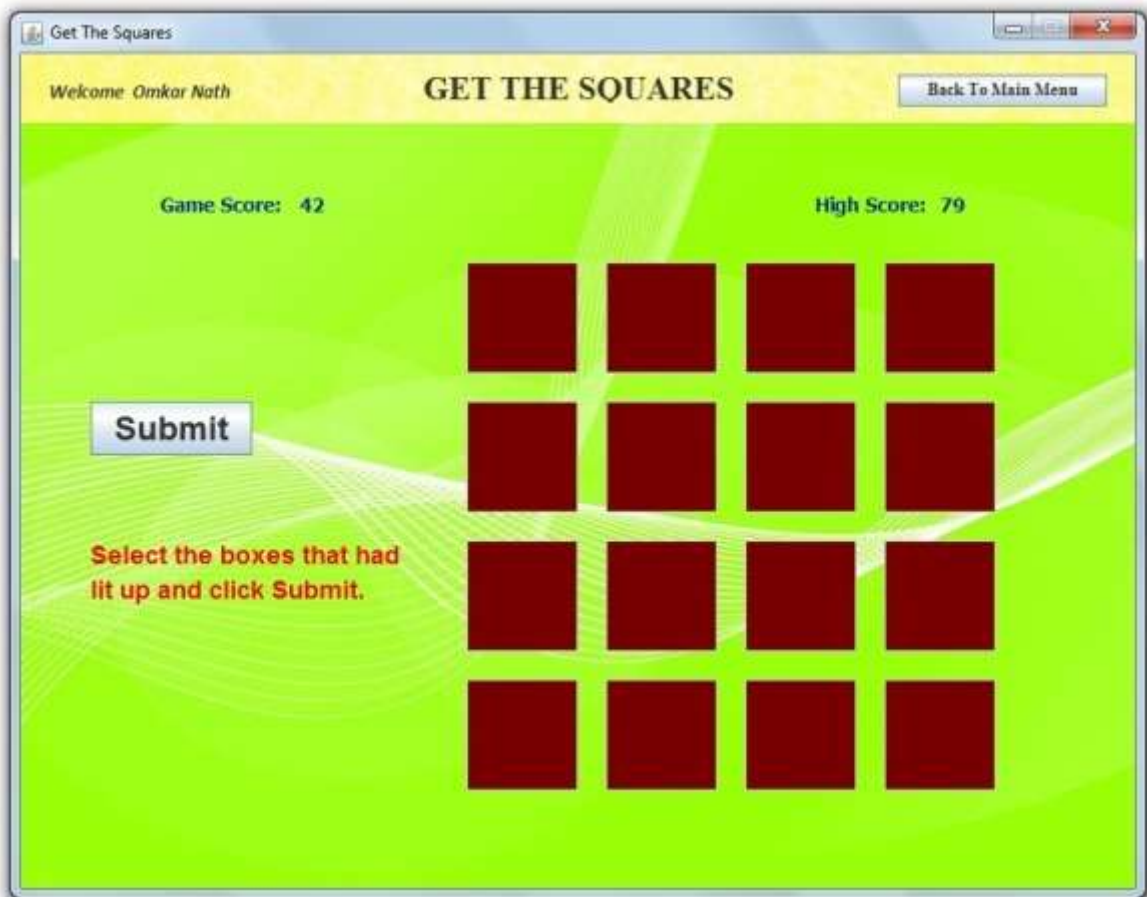
    correctMsgL.setVisible(true);

    playAudio(5);

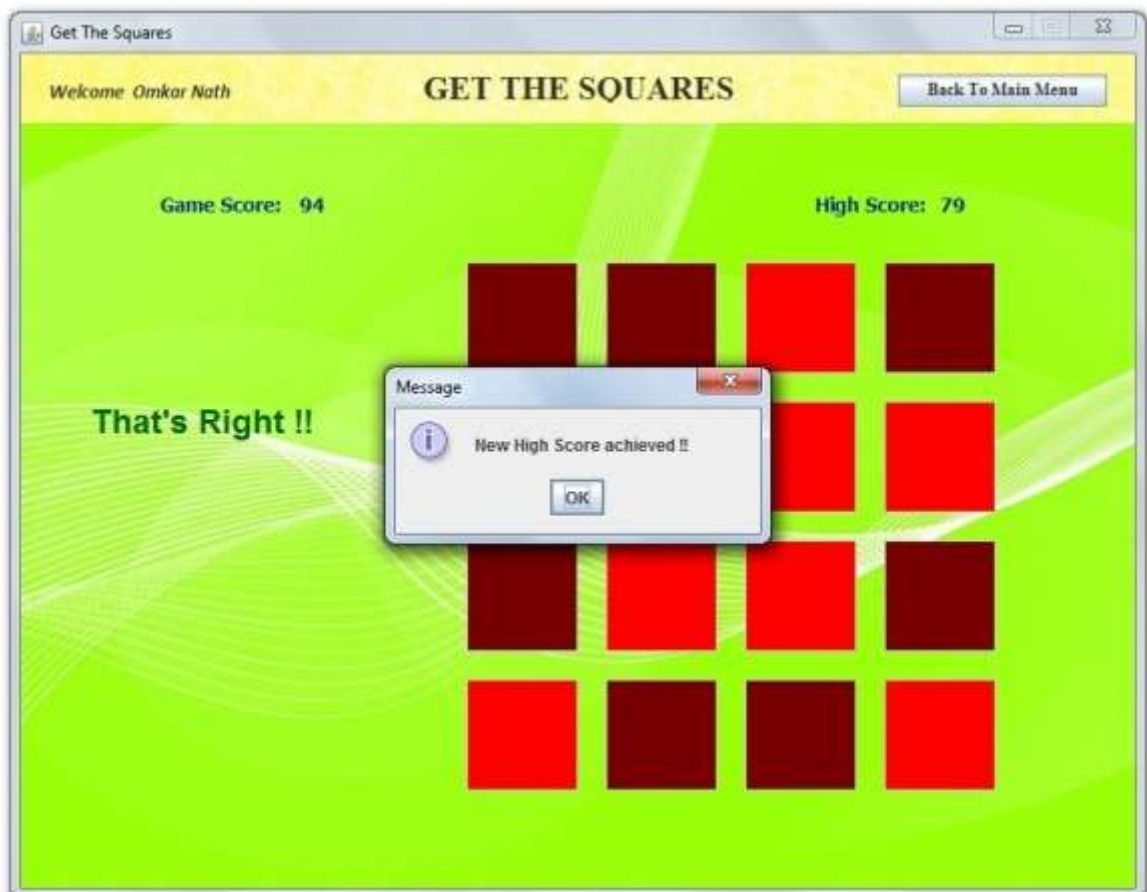
    temp = ((8 - Math.abs(numberOfBoxes - 8)));
    if (temp == 0) {
        temp++;
    }
    if ((numberOfBoxes >= 5) && (numberOfBoxes <= 12)) {
        temp += Math.round((float) (temp * 0.5));
    }
    gameScore += Math.round(temp * (((float) (msTimeMaximum - msTimeInterval) / (float)
(msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
    gameScore2L.setText(Integer.toString(gameScore));

    if (gameScore > highScore) {
        if (!newHighScore && highScore != 0) {
            newHighScore = true;
            playAudio(6);
            JOptionPane.showMessageDialog(this, "New High Score achieved !!");
        }
        highScore = gameScore;
        highScore2L.setText(Integer.toString(highScore));

        try {
            Class.forName("java.sql.DriverManager");
```



Waiting for Player to Select Boxes



Message on achieving New High Score

```
        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
        Statement stmt = (Statement) con.createStatement();
        String query = "UPDATE highscores SET GetTheSquares = " + highScore + " WHERE
Username = " + username + ";";
        stmt.executeUpdate(query);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
}

msTimeInterval -= msTimeStep;
if (msTimeInterval < msTimeMinimum) {
    msTimeInterval = msTimeMinimum;
}

nextLevel(msTimeEnd);
}
}
```

### Text Field – 1:

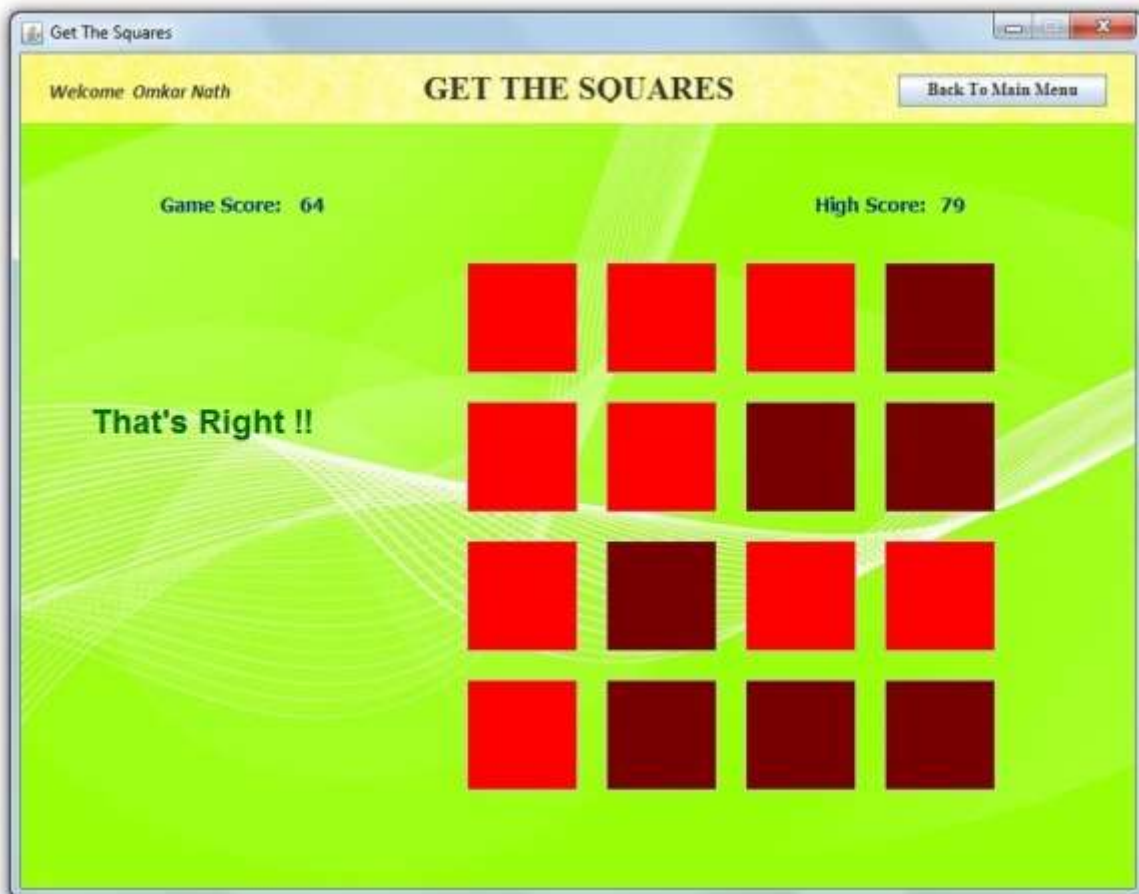
```
private void box01TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[1]) {
        setBoxHighlighted(1);
    }
}

private void box01TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[1]) {
        setBoxNormal(1);
    }
}

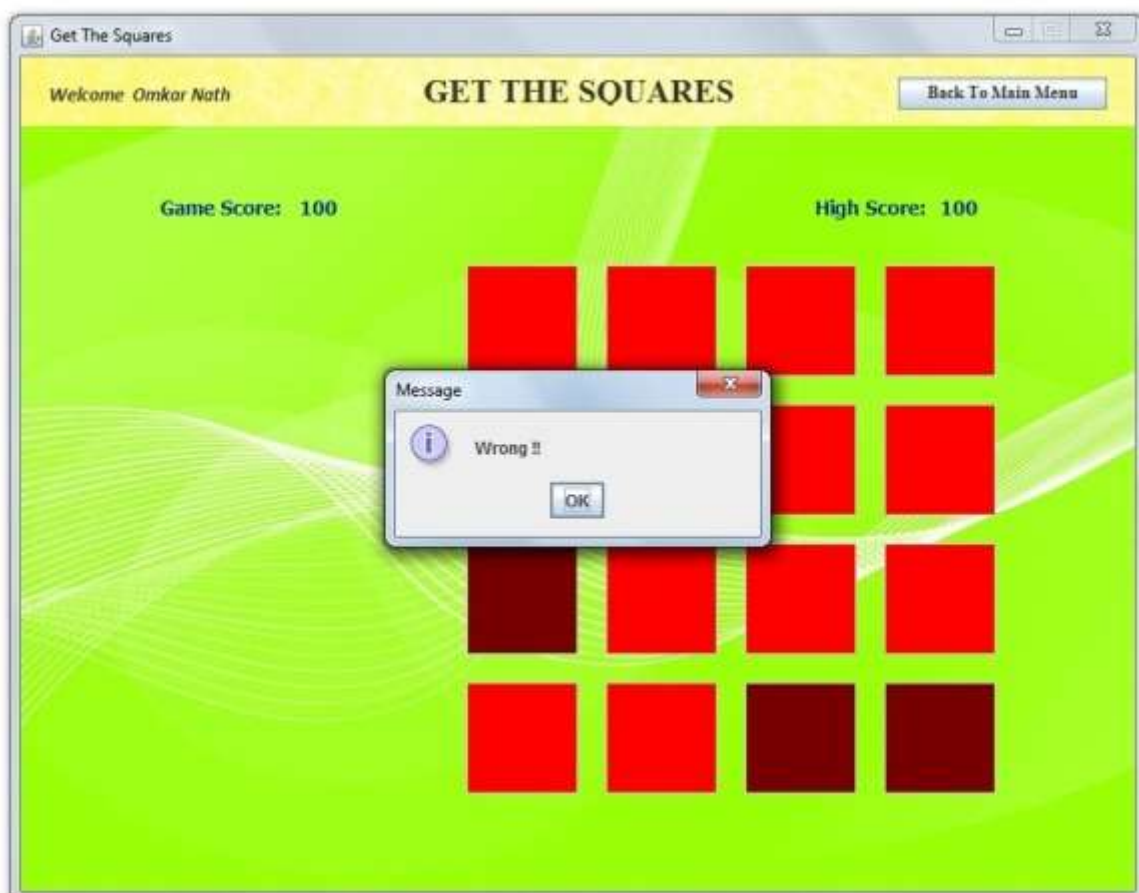
private void box01TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(1);
    }
}

private void box01TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(1);
    }
}

private void box01TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[1] = !selected[1];
    }
}
```



Message on getting it Right



Message on getting it Wrong

## Text Field – 2:

```
private void box02TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[2] = !selected[2];
    }
}

private void box02TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[2]) {
        setBoxHighlighted(2);
    }
}

private void box02TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[2]) {
        setBoxNormal(2);
    }
}

private void box02TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(2);
    }
}

private void box02TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(2);
    }
}
```

## Text Field – 3:

```
private void box03TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[3] = !selected[3];
    }
}

private void box03TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[3]) {
        setBoxHighlighted(3);
    }
}

private void box03TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[3]) {
        setBoxNormal(3);
    }
}

private void box03TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
```

```
        setBoxPressed(3);
    }
}

private void box03TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(3);
    }
}
```

#### Text Field – 4:

```
private void box04TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[4] = !selected[4];
    }
}

private void box04TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[4]) {
        setBoxHighlighted(4);
    }
}

private void box04TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[4]) {
        setBoxNormal(4);
    }
}

private void box04TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(4);
    }
}

private void box04TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(4);
    }
}
```

#### Text Field – 5:

```
private void box05TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[5] = !selected[5];
    }
}

private void box05TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[5]) {
        setBoxHighlighted(5);
    }
}
```

```
}

private void box05TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[5]) {
        setBoxNormal(5);
    }
}

private void box05TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(5);
    }
}

private void box05TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(5);
    }
}
```

### Text Field – 6:

```
private void box06TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[6] = !selected[6];
    }
}

private void box06TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[6]) {
        setBoxHighlighted(6);
    }
}

private void box06TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[6]) {
        setBoxNormal(6);
    }
}

private void box06TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(6);
    }
}

private void box06TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(6);
    }
}
```

**Text Field – 7:**

```
private void box07TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[7] = !selected[7];
    }
}

private void box07TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[7]) {
        setBoxHighlighted(7);
    }
}

private void box07TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[7]) {
        setBoxNormal(7);
    }
}

private void box07TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(7);
    }
}

private void box07TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(7);
    }
}
```

**Text Field – 8:**

```
private void box08TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[8] = !selected[8];
    }
}

private void box08TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[8]) {
        setBoxHighlighted(8);
    }
}

private void box08TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[8]) {
        setBoxNormal(8);
    }
}

private void box08TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
```



```
        setBoxPressed(8);
    }
}

private void box08TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(8);
    }
}
```

### Text Field – 9:

```
private void box09TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[9] = !selected[9];
    }
}

private void box09TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[9]) {
        setBoxHighlighted(9);
    }
}

private void box09TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[9]) {
        setBoxNormal(9);
    }
}

private void box09TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(9);
    }
}

private void box09TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(9);
    }
}
```

### Text Field – 10:

```
private void box10TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[10] = !selected[10];
    }
}

private void box10TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[10]) {
        setBoxHighlighted(10);
    }
}
```

```
}

private void box10TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[10]) {
        setBoxNormal(10);
    }
}

private void box10TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(10);
    }
}

private void box10TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(10);
    }
}
```

### Text Field – 11:

```
private void box11TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[11] = !selected[11];
    }
}

private void box11TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[11]) {
        setBoxHighlighted(11);
    }
}

private void box11TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[11]) {
        setBoxNormal(11);
    }
}

private void box11TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(11);
    }
}

private void box11TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(11);
    }
}
```

**Text Field – 12:**

```
private void box12TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[12] = !selected[12];
    }
}

private void box12TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[12]) {
        setBoxHighlighted(12);
    }
}

private void box12TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[12]) {
        setBoxNormal(12);
    }
}

private void box12TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(12);
    }
}

private void box12TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(12);
    }
}
```

**Text Field – 13:**

```
private void box13TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[13] = !selected[13];
    }
}

private void box13TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[13]) {
        setBoxHighlighted(13);
    }
}

private void box13TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[13]) {
        setBoxNormal(13);
    }
}

private void box13TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
```

```
        setBoxPressed(13);
    }
}

private void box13TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(13);
    }
}
```

### Text Field – 14:

```
private void box14TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[14] = !selected[14];
    }
}

private void box14TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[14]) {
        setBoxHighlighted(14);
    }
}

private void box14TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[14]) {
        setBoxNormal(14);
    }
}

private void box14TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(14);
    }
}

private void box14TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(14);
    }
}
```

### Text Field – 15:

```
private void box15TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[15] = !selected[15];
    }
}

private void box15TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[15]) {
        setBoxHighlighted(15);
    }
}
```

```
}

private void box15TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[15]) {
        setBoxNormal(15);
    }
}

private void box15TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(15);
    }
}

private void box15TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(15);
    }
}
```

### Text Field – 16:

```
private void box16TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable) {
        selected[16] = !selected[16];
    }
}

private void box16TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[16]) {
        setBoxHighlighted(16);
    }
}

private void box16TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable && !selected[16]) {
        setBoxNormal(16);
    }
}

private void box16TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxPressed(16);
    }
}

private void box16TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable) {
        setBoxHighlighted(16);
    }
}
```

## Game Over:

### Button – Play Again:

```
private void playAgainBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    gameOverPanel.setVisible(false);  
  
    exit = true;  
  
    clickable = false;  
  
    msTimeInterval = msTimeMaximum;  
  
    gameScore = 0;  
    gameScore2L.setText(Integer.toString(gameScore));  
  
    highScore2L.setText(Integer.toString(highScore));  
  
    if (highScore == 0) {  
        newHighScore = true;  
    } else {  
        newHighScore = false;  
    }  
  
    startBtn.setText("Start Game");  
    startBtn.setVisible(true);  
  
    startMsg1L.setVisible(true);  
    startMsg2L.setVisible(true);  
  
    submitBtn.setVisible(false);  
    submitMsg1L.setVisible(false);  
    submitMsg2L.setVisible(false);  
    correctMsgL.setVisible(false);  
  
    for (int i = 1; i <= 16; i++) {  
        setBoxNormal(i);  
    }  
  
    gamePanel.setVisible(true);  
  
    playAudio(1);  
}
```



End of Game

## **Frame: Match the Colours (Game 3)**

### **Importing Java Packages:**

```
import com.mysql.jdbc.Connection;  
import com.mysql.jdbc.Statement;  
import java.sql.DriverManager;
```

```
import java.awt.*;
```

```
import java.io.File;  
import java.io.FileInputStream;  
import java.io.InputStream;  
import sun.audio.AudioPlayer;  
import sun.audio.AudioStream;
```

```
import java.util.*;
```

```
import javax.swing.JOptionPane;
```

### **Global Variables:**

```
Timer timerDelay;  
Timer timerDisplay;  
Timer timerCorrect;
```

```
int msTimeBegin = 750;  
int msTimeEnd = 1500;  
int msTimeMaximum = 2200;  
int msTimeMinimum = 200;  
int msTimeStep = 200;  
int msTimeInterval;  
int maxMultiple = 3;
```

```
String colourNormal = "0x777777";  
String colourPressed = "0x999999";  
String colourHighlighted = "0xAAAAAA";  
String colourRed = "0xFF0000";  
String colourGreen = "0x00FF00";  
String colourBlue = "0x0000FF";  
String colourYellow = "0xFFFF00";  
String username = "";
```

```
int gameScore = 0;  
int highScore;  
int control = 0;  
int count = 0;
```

```
int[] config = new int[9];
```

```
boolean newHighScore;
```



```
boolean begin = true;
boolean exit;

boolean[] clickable = new boolean[9];
```

## Event – Form Window Activated:

```
private void formWindowActivated(java.awt.event.WindowEvent evt) {
    if (begin) {
        msTimeInterval = msTimeMaximum;

        if (highScore == 0) {
            newHighScore = false;
        } else {
            newHighScore = true;
        }

        hScore2L.setText("" + highScore);
        gameScore2L.setText("" + gameScore);

        setAllBoxesNormal();

        startMsg1L.setVisible(true);
        startMsg2L.setVisible(true);
        startMsg3L.setVisible(true);

        correctMsgL.setVisible(false);
        instructionsMsg1L.setVisible(false);
        instructionsMsg2L.setVisible(false);

        gamePanel.setVisible(false);
        gameOverPanel.setVisible(false);

        coverPanel.setVisible(false);

        begin = false;
        exit = true;
    }
}
```

## Methods & Functions Used:

```
public void display(int msDelay, int msFlash) {
    timerDelay = new Timer();
    timerDisplay = new Timer();
    timerDelay.schedule(new taskDelay(), msDelay);
    timerDisplay.schedule(new taskDisplay(), msFlash);
}

class taskDelay extends TimerTask {
    public void run() {
        int i;
        playAudio(3);
    }
}
```

```
        for (i = 1; i <= 8; i++) {
            setBoxColour(i, config[i]);
        }
        timerDelay.cancel();
    }
}

class taskDisplay extends TimerTask {
    public void run() {
        setAllBoxesNormal();
        for (int i = 1; i <= 8; i++) {
            clickable[i] = true;
        }
        instructionsMsg1L.setVisible(true);
        instructionsMsg2L.setVisible(true);
        timerDisplay.cancel();
    }
}

public void gameOver() {
    for (int i = 1; i <= 8; i++) {
        setBoxColour(i, config[i]);
    }

    instructionsMsg1L.setVisible(false);
    instructionsMsg2L.setVisible(false);

    playAudio(7);
    JOptionPane.showMessageDialog(this, "Wrong !!");
    setAllBoxesNormal();

    count = 0;

    exit = true;

    gameScoreL.setText("\nGame Score: " + gameScore);
    highScoreL.setText("\nHigh Score: " + highScore);

    gamePanel.setVisible(false);
    gameOverPanel.setVisible(true);
}

public void newLevel() {
    count = 0;

    msTimeInterval -= msTimeStep;
    if (msTimeInterval < msTimeMinimum) {
        msTimeInterval = msTimeMinimum;
    }

    instructionsMsg1L.setVisible(false);
    instructionsMsg2L.setVisible(false);
    correctMsgL.setVisible(true);

    nextLevel(msTimeEnd);
}
```

```
}

public void nextLevel(int msTime) {
    timerCorrect = new Timer();
    timerCorrect.schedule(new taskCorrect(), msTime);
}

class taskCorrect extends TimerTask {
    public void run() {
        setAllBoxesNormal();
        correctMsgL.setVisible(false);
        startBtn.setText("Start");
        startBtn.setVisible(true);
        startMsg1L.setVisible(true);
        startMsg2L.setVisible(true);
        startMsg3L.setVisible(true);
    }
}

public void setAllBoxesNormal() {
    for (int i = 1; i <= 8; i++) {
        setBoxNormal(i);
    }
}

public void setBoxNormal(int box) {
    switch (box) {
        case 1:
            box1TF.setBackground(Color.decode(colourNormal));
            break;
        case 2:
            box2TF.setBackground(Color.decode(colourNormal));
            break;
        case 3:
            box3TF.setBackground(Color.decode(colourNormal));
            break;
        case 4:
            box4TF.setBackground(Color.decode(colourNormal));
            break;
        case 5:
            box5TF.setBackground(Color.decode(colourNormal));
            break;
        case 6:
            box6TF.setBackground(Color.decode(colourNormal));
            break;
        case 7:
            box7TF.setBackground(Color.decode(colourNormal));
            break;
        case 8:
            box8TF.setBackground(Color.decode(colourNormal));
            break;
    }
}
```

```
public void setBoxHighlighted(int box) {
    switch (box) {
        case 1:
            box1TF.setBackground(Color.decode(colourHighlighted));
            break;
        case 2:
            box2TF.setBackground(Color.decode(colourHighlighted));
            break;
        case 3:
            box3TF.setBackground(Color.decode(colourHighlighted));
            break;
        case 4:
            box4TF.setBackground(Color.decode(colourHighlighted));
            break;
        case 5:
            box5TF.setBackground(Color.decode(colourHighlighted));
            break;
        case 6:
            box6TF.setBackground(Color.decode(colourHighlighted));
            break;
        case 7:
            box7TF.setBackground(Color.decode(colourHighlighted));
            break;
        case 8:
            box8TF.setBackground(Color.decode(colourHighlighted));
            break;
    }
}
```

```
public void setBoxPressed(int box) {
    switch (box) {
        case 1:
            box1TF.setBackground(Color.decode(colourPressed));
            break;
        case 2:
            box2TF.setBackground(Color.decode(colourPressed));
            break;
        case 3:
            box3TF.setBackground(Color.decode(colourPressed));
            break;
        case 4:
            box4TF.setBackground(Color.decode(colourPressed));
            break;
        case 5:
            box5TF.setBackground(Color.decode(colourPressed));
            break;
        case 6:
            box6TF.setBackground(Color.decode(colourPressed));
            break;
        case 7:
            box7TF.setBackground(Color.decode(colourPressed));
            break;
        case 8:
            box8TF.setBackground(Color.decode(colourPressed));
            break;
    }
}
```

```
    }  
}  
  
public void setBoxColour(int box, int colour) {  
    switch (colour) {  
        case 1:  
            setBoxColourString(box, colourRed);  
            break;  
        case 2:  
            setBoxColourString(box, colourGreen);  
            break;  
        case 3:  
            setBoxColourString(box, colourBlue);  
            break;  
        case 4:  
            setBoxColourString(box, colourYellow);  
            break;  
    }  
}  
  
public void setBoxColourString(int nBox, String sColour) {  
    switch (nBox) {  
        case 1:  
            box1TF.setBackground(Color.decode(sColour));  
            break;  
        case 2:  
            box2TF.setBackground(Color.decode(sColour));  
            break;  
        case 3:  
            box3TF.setBackground(Color.decode(sColour));  
            break;  
        case 4:  
            box4TF.setBackground(Color.decode(sColour));  
            break;  
        case 5:  
            box5TF.setBackground(Color.decode(sColour));  
            break;  
        case 6:  
            box6TF.setBackground(Color.decode(sColour));  
            break;  
        case 7:  
            box7TF.setBackground(Color.decode(sColour));  
            break;  
        case 8:  
            box8TF.setBackground(Color.decode(sColour));  
            break;  
    }  
}  
  
public void dataPass3(String uName, int sc) {  
    username = uName;  
    highScore = sc;  
    nameL.setText(username);  
    hScore2L.setText("" + highScore);  
}
```

```
public void playAudio(int track) {
    switch (track) {
        case 0:
            Toolkit.getDefaultToolkit().beep();
            break;
        case 1:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\proceedToGame.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 2:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\singleBeep.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 3:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\glassPing.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 4:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\singleCoinDrop.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 5:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\multipleCoinDrop.wav"));
                AudioStream audio = new AudioStream(music);
                AudioPlayer.player.start(audio);
            } catch (Exception e) {
                JOptionPane.showMessageDialog(this, e.getMessage());
            }
            break;
        case 6:
            try {
                InputStream music = new FileInputStream(new File("src\\Audio\\tadaHighScore.wav"));
                AudioStream audio = new AudioStream(music);
            }
    }
}
```

```

        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
case 7:
    try {
        InputStream music = new FileInputStream(new File("src\\Audio\\buzzerGameOver.wav"));
        AudioStream audio = new AudioStream(music);
        AudioPlayer.player.start(audio);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
    break;
}
}
}

```

## Instructions:

### Button – Back To Main Menu:

```

private void returnBtnActionPerformed(java.awt.event.ActionEvent evt) {
    if (!exit) {
        playAudio(0);
        int reply = JOptionPane.showConfirmDialog(this, "Game In Progress !!\nAre you sure you want to
exit", "Caution !!", JOptionPane.YES_NO_OPTION);
        if (reply == JOptionPane.YES_OPTION) {
            exit = true;
        }
    }

    if (exit) {
        homePage frame0 = new homePage();
        matchTheColours.this.setVisible(false);
        frame0.setVisible(true);
        frame0.dataPass0(username);
    }
}

```

### Button – Proceed To Game:

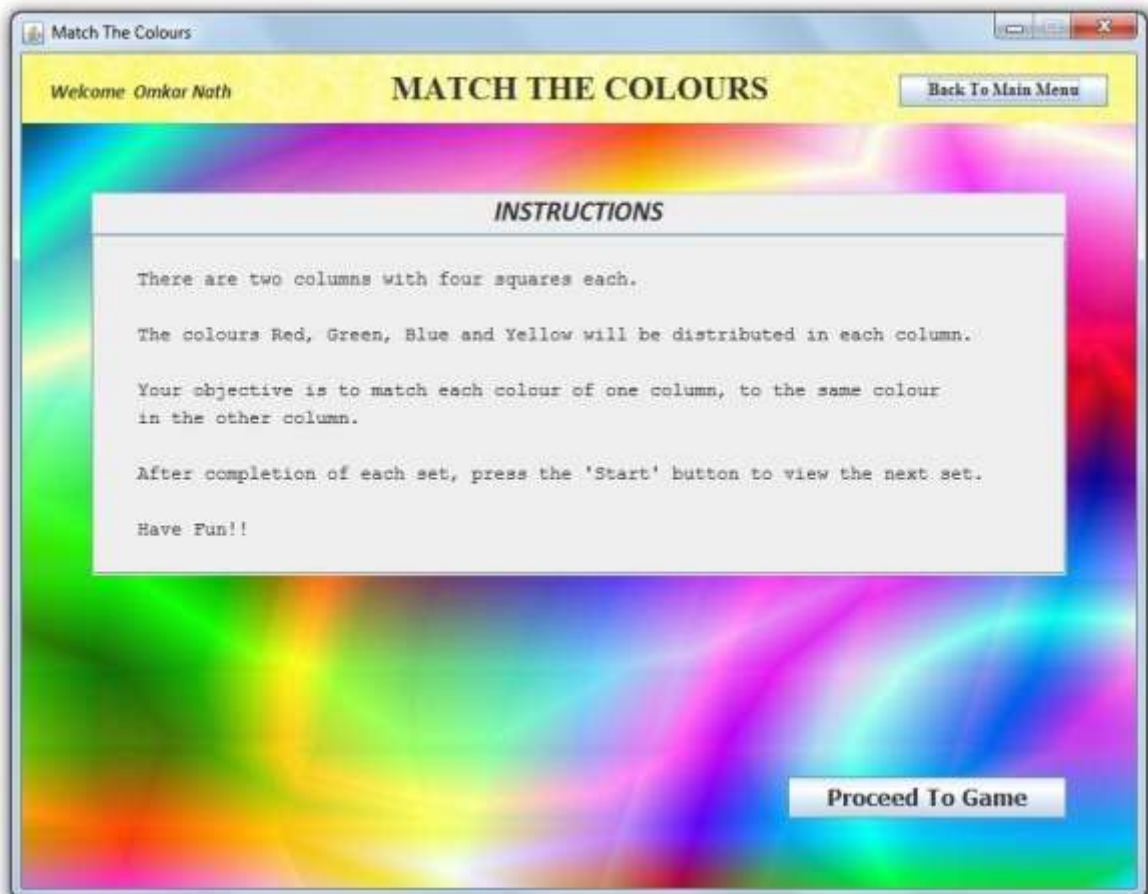
```

private void proceedToGameBtnActionPerformed(java.awt.event.ActionEvent evt) {
    instructionsPanel.setVisible(false);

    gamePanel.setVisible(true);

    playAudio(1);
}

```



Instructions of the Game



## Game:

### Button – Start:

```
private void startBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    int i, j, flag = 0;  
  
    startBtn.setVisible(false);  
  
    startMsg1L.setVisible(false);  
    startMsg2L.setVisible(false);  
    startMsg3L.setVisible(false);  
  
    exit = false;  
  
    for (i = 1; i <= 8; i++) {  
        config[i] = 0;  
        clickable[i] = false;  
    }  
  
    for (i = 1; i <= 4; i++) {  
        flag = 0;  
        while (flag == 0) {  
            j = (int) (Math.random() * 4 + 1);  
            if (config[j] == 0) {  
                config[j] = i;  
                flag = 1;  
            }  
        }  
    }  
  
    for (i = 1; i <= 4; i++) {  
        flag = 0;  
        while (flag == 0) {  
            j = (int) (Math.random() * 4 + 5);  
            if (config[j] == 0) {  
                config[j] = i;  
                flag = 1;  
            }  
        }  
    }  
  
    display(msTimeBegin, msTimeBegin + msTimeInterval);  
}
```

### Text Field – 1:

```
private void box1TFMouseEntered(java.awt.event.MouseEvent evt) {  
    if (clickable[1]) {  
        setBoxHighlighted(1);  
    }  
}
```



Starting the Game



Starting a new Level

```

private void box1TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable[1]) {
        setBoxNormal(1);
    }
}

private void box1TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable[1]) {
        setBoxPressed(1);
    }
}

private void box1TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable[1]) {
        setBoxHighlighted(1);
    }
}

private void box1TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable[1]) {
        clickable[1] = false;
        count = count + 1;
        setBoxColour(1, config[1]);
        if (control == 0) {
            control = config[1];
        } else {
            if (control == config[1]) {
                control = 0;
            }

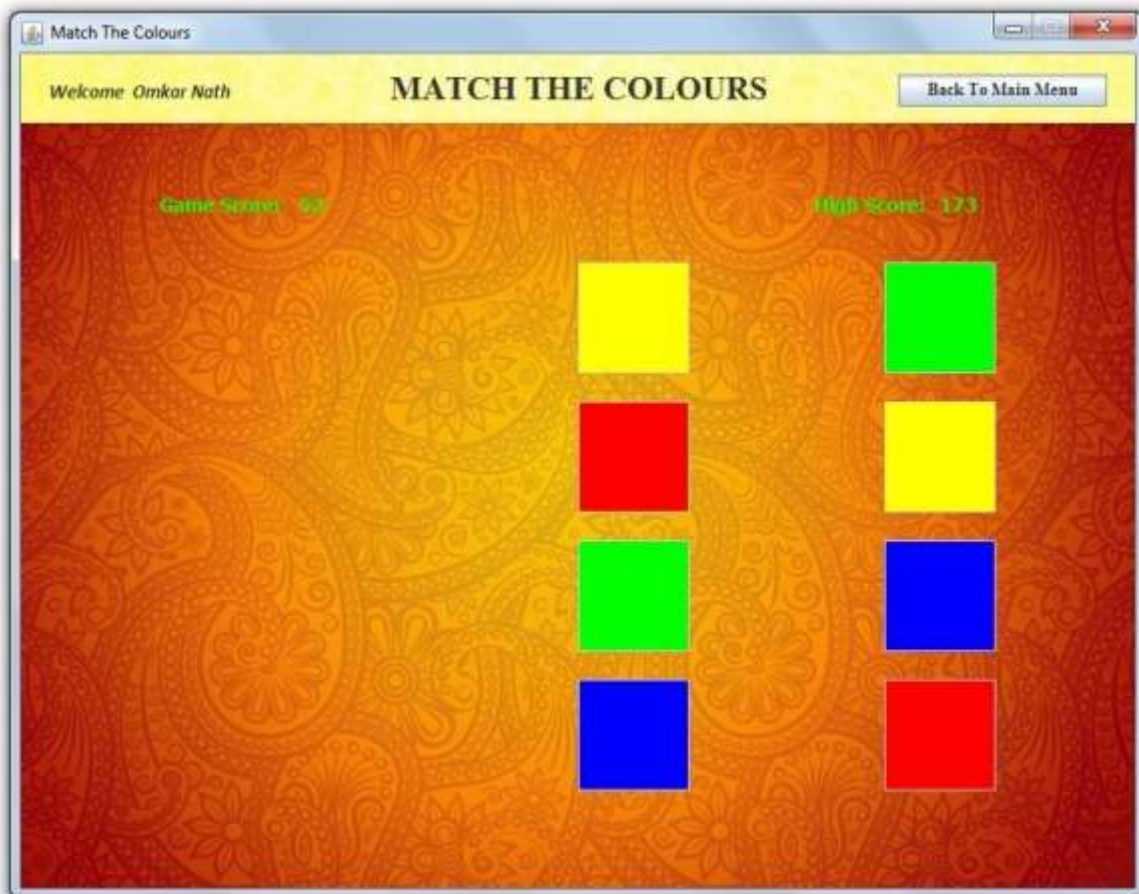
            playAudio(4);
            gameScore += Math.round((5 - (float) (count / 2)) * (((float) (msTimeMaximum -
msTimeInterval) / (float) (msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
            gameScore2L.setText("" + gameScore);

            if (gameScore > highScore) {
                if (newHighScore && highScore != 0) {
                    playAudio(6);
                    JOptionPane.showMessageDialog(this, "New High Score Acheived !!");
                    newHighScore = false;
                }

                highScore = gameScore;
                hScore2L.setText("" + highScore);

                try {
                    Class.forName("java.sql.DriverManager");
                    Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
                    Statement stmt = (Statement) con.createStatement();
                    String query = "UPDATE highscores SET MatchTheColours = " + highScore + " WHERE
Username = " + username + """;
                    stmt.executeUpdate(query);
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(this, e.getMessage());
                }
            }
        }
    }
}

```



Auto Generated Random Colour Set



Waiting for Player to Select



```

        }
    }

    if (count == 8) {
        newLevel();
    }
    else {
        gameOver();
        control = 0;
    }
}
}
}

```

## Text Field – 2:

```

private void box2TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable[2]) {
        setBoxHighlighted(2);
    }
}

private void box2TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable[2]) {
        setBoxNormal(2);
    }
}

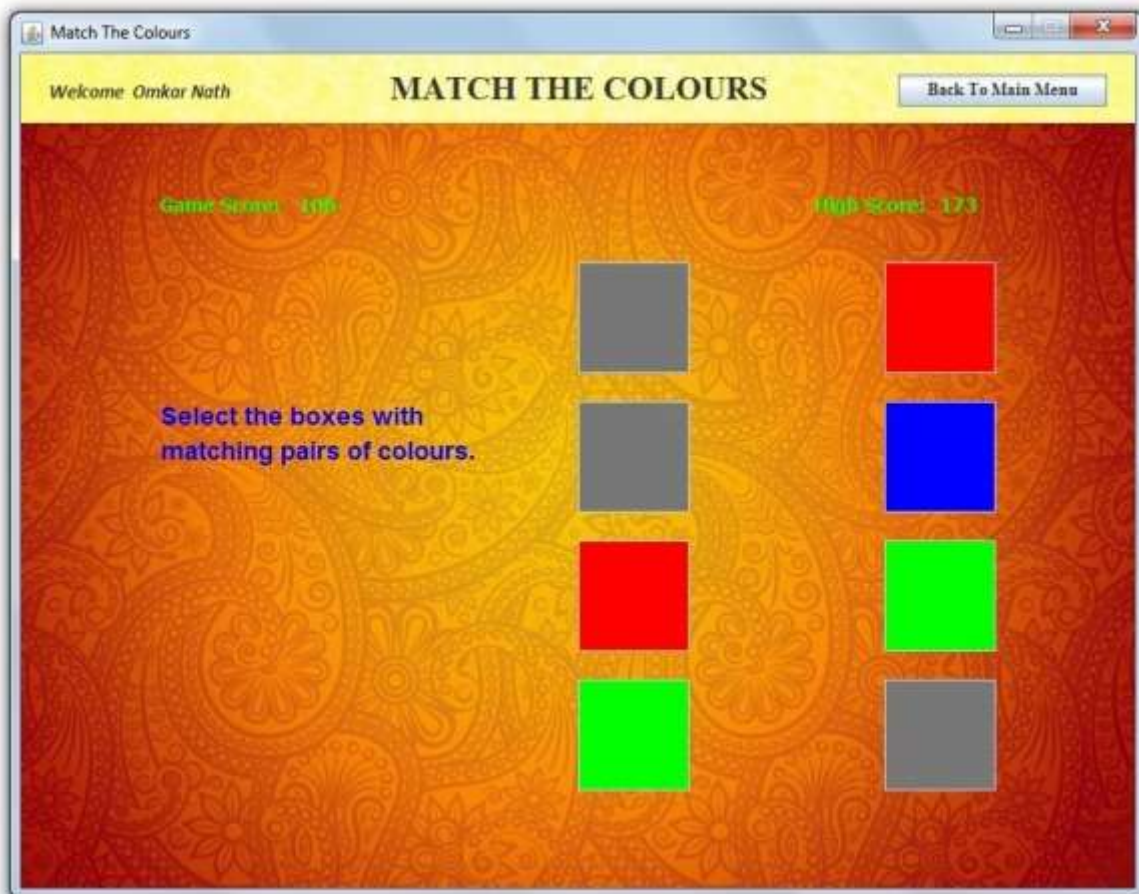
private void box2TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable[2]) {
        setBoxPressed(2);
    }
}

private void box2TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable[2]) {
        setBoxHighlighted(2);
    }
}

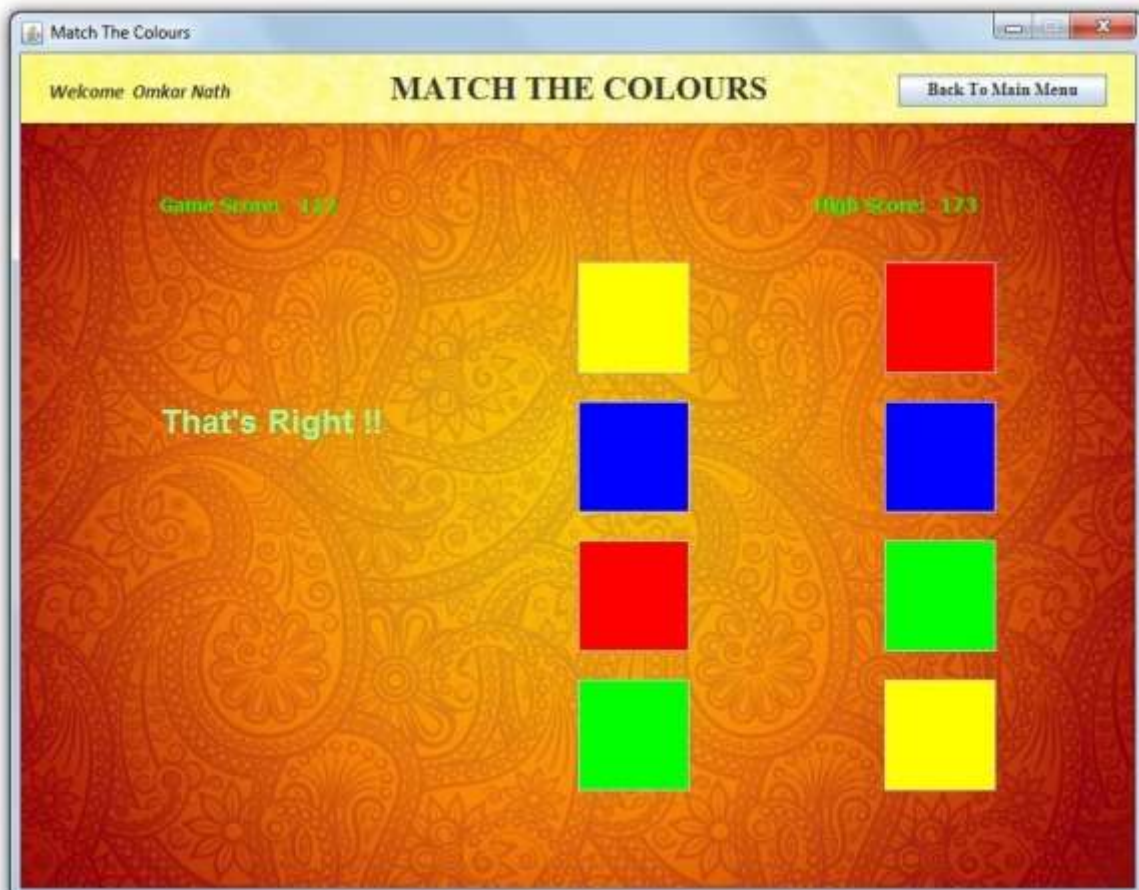
private void box2TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable[2]) {
        clickable[2] = false;
        count = count + 1;
        setBoxColour(2, config[2]);
        if (control == 0) {
            control = config[2];
        } else {
            if (control == config[2]) {
                control = 0;
            }
        }

        playAudio(4);
        gameScore += Math.round((5 - (float) (count / 2)) * (((float) (msTimeMaximum -
msTimeInterval) / (float) (msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
    }
}

```

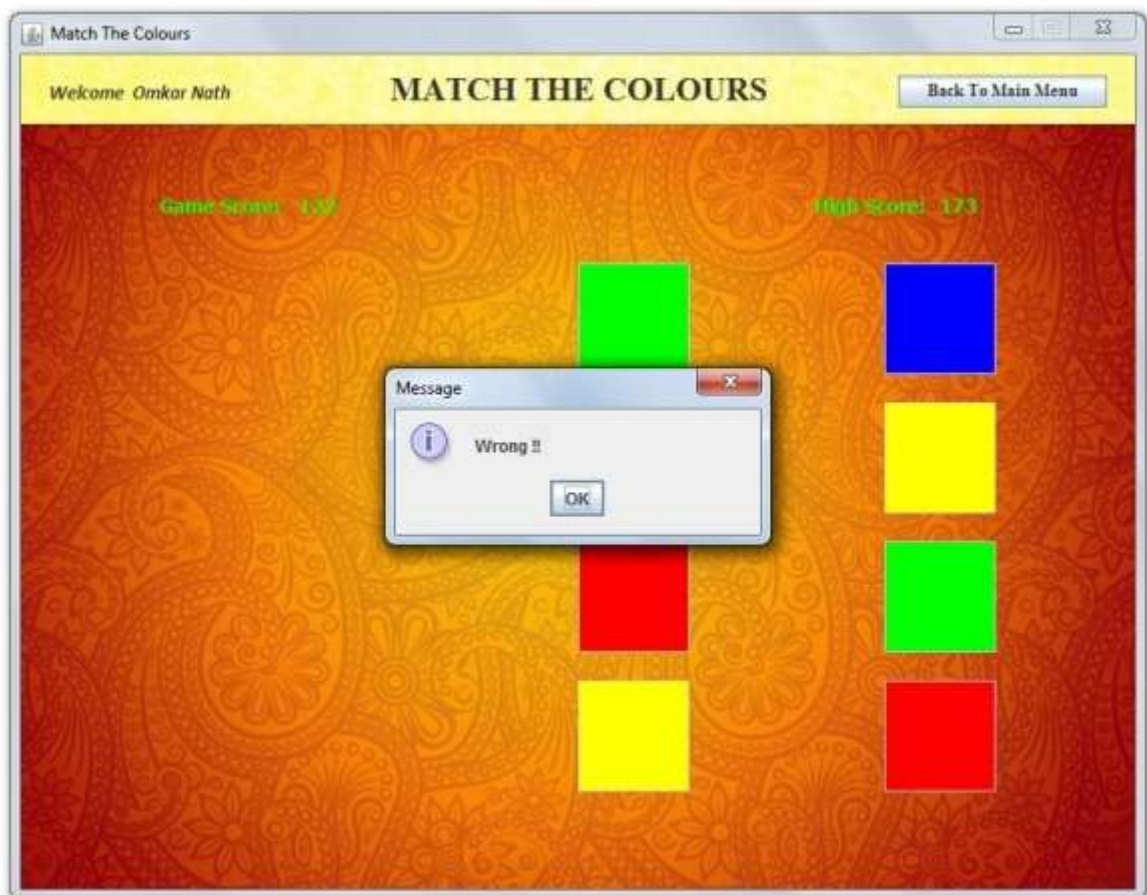


Player in the Middle of Selecting



Message on getting it Right





Message on getting it Wrong



```

private void box3TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable[3]) {
        setBoxHighlighted(3);
    }
}

private void box3TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable[3]) {
        clickable[3] = false;
        count = count + 1;
        setBoxColour(3, config[3]);
        if (control == 0) {
            control = config[3];
        } else {
            if (control == config[3]) {
                control = 0;

                playAudio(4);
                gameScore += Math.round((5 - (float) (count / 2)) * (((float) (msTimeMaximum -
msTimeInterval) / (float) (msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
                gameScore2L.setText("" + gameScore);

                if (gameScore > highScore) {
                    if (newHighScore && highScore != 0) {
                        playAudio(6);
                        JOptionPane.showMessageDialog(this, "New High Score Acheived !!");
                        newHighScore = false;
                    }

                    highScore = gameScore;
                    hScore2L.setText("" + highScore);

                    try {
                        Class.forName("java.sql.DriverManager");
                        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
                        Statement stmt = (Statement) con.createStatement();
                        String query = "UPDATE highscores SET MatchTheColours = " + highScore + " WHERE
Username = " + username + " ";
                        stmt.executeUpdate(query);
                    } catch (Exception e) {
                        JOptionPane.showMessageDialog(this, e.getMessage());
                    }
                }

                if (count == 8) {
                    newLevel();
                }
            } else {
                gameOver();
            }
        }
    }
}

```

**Text Field – 4:**

```
private void box4TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable[4]) {
        setBoxHighlighted(4);
    }
}

private void box4TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable[4]) {
        setBoxNormal(4);
    }
}

private void box4TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable[4]) {
        setBoxPressed(4);
    }
}

private void box4TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable[4]) {
        setBoxHighlighted(4);
    }
}

private void box4TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable[4]) {
        clickable[4] = false;
        count = count + 1;
        setBoxColour(4, config[4]);
        if (control == 0) {
            control = config[4];
        } else {
            if (control == config[4]) {
                control = 0;
            }
        }

        playAudio(4);
        gameScore += Math.round(((5 - (float) (count / 2)) * (((float) (msTimeMaximum -
msTimeInterval) / (float) (msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
        gameScore2L.setText("" + gameScore);

        if (gameScore > highScore) {
            if (newHighScore && highScore != 0) {
                playAudio(6);
                JOptionPane.showMessageDialog(this, "New High Score Acheived !!");
                newHighScore = false;
            }
        }

        highScore = gameScore;
        hScore2L.setText("" + highScore);

        try {
            Class.forName("java.sql.DriverManager");
        }
    }
}
```

```

        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
        Statement stmt = (Statement) con.createStatement();
        String query = "UPDATE highscores SET MatchTheColours = " + highScore + " WHERE
Username = " + username + ",";
        stmt.executeUpdate(query);
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
    }

    if (count == 8) {
        newLevel();
    }
    } else {
        gameOver();
    }
}
}
}
}

```

### Text Field – 5:

```

private void box5TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable[5]) {
        setBoxHighlighted(5);
    }
}

private void box5TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable[5]) {
        setBoxNormal(5);
    }
}

private void box5TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable[5]) {
        setBoxPressed(5);
    }
}

private void box5TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable[5]) {
        setBoxHighlighted(5);
    }
}

private void box5TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable[5]) {
        clickable[5] = false;
        count = count + 1;
        setBoxColour(5, config[5]);
        if (control == 0) {
            control = config[5];

```

```

    } else {
        if (control == config[5]) {
            control = 0;

            playAudio(4);
            gameScore += Math.round((5 - (float) (count / 2)) * (((float) (msTimeMaximum -
msTimeInterval) / (float) (msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
            gameScore2L.setText("" + gameScore);

            if (gameScore > highScore) {
                if (newHighScore && highScore != 0) {
                    playAudio(6);
                    JOptionPane.showMessageDialog(this, "New High Score Acheived !!");
                    newHighScore = false;
                }

                highScore = gameScore;
                hScore2L.setText("" + highScore);

                try {
                    Class.forName("java.sql.DriverManager");
                    Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
                    Statement stmt = (Statement) con.createStatement();
                    String query = "UPDATE highscores SET MatchTheColours = " + highScore + " WHERE
Username = " + username + " ";
                    stmt.executeUpdate(query);
                } catch (Exception e) {
                    JOptionPane.showMessageDialog(this, e.getMessage());
                }
            }

            if (count == 8) {
                newLevel();
            }
        } else {
            gameOver();
        }
    }
}

```

### Text Field – 6:

```

private void box6TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable[6]) {
        setBoxHighlighted(6);
    }
}

private void box6TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable[6]) {
        setBoxNormal(6);
    }
}

```

```

    }

    private void box6TFMousePressed(java.awt.event.MouseEvent evt) {
        if (clickable[6]) {
            setBoxPressed(6);
        }
    }

    private void box6TFMouseReleased(java.awt.event.MouseEvent evt) {
        if (clickable[6]) {
            setBoxHighlighted(6);
        }
    }

    private void box6TFMouseClicked(java.awt.event.MouseEvent evt) {
        if (clickable[6]) {
            clickable[6] = false;
            count = count + 1;
            setBoxColour(6, config[6]);
            if (control == 0) {
                control = config[6];
            } else {
                if (control == config[6]) {
                    control = 0;
                }

                playAudio(4);
                gameScore += Math.round((5 - (float) (count / 2)) * (((float) (msTimeMaximum -
msTimeInterval) / (float) (msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
                gameScore2L.setText("" + gameScore);

                if (gameScore > highScore) {
                    if (newHighScore && highScore != 0) {
                        playAudio(6);
                        JOptionPane.showMessageDialog(this, "New High Score Acheived !!");
                        newHighScore = false;
                    }

                    highScore = gameScore;
                    hScore2L.setText("" + highScore);

                    try {
                        Class.forName("java.sql.DriverManager");
                        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
                        Statement stmt = (Statement) con.createStatement();
                        String query = "UPDATE highscores SET MatchTheColours = " + highScore + " WHERE
Username = " + username + " ";
                        stmt.executeUpdate(query);
                    } catch (Exception e) {
                        JOptionPane.showMessageDialog(this, e.getMessage());
                    }
                }

                if (count == 8) {
                    newLevel();
                }
            }
        }
    }

```

```

    }
    } else {
        gameOver();
    }
}
}
}

```

### Text Field – 7:

```

private void box7TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable[7]) {
        setBoxHighlighted(7);
    }
}

```

```

private void box7TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable[7]) {
        setBoxNormal(7);
    }
}

```

```

private void box7TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable[7]) {
        setBoxPressed(7);
    }
}

```

```

private void box7TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable[7]) {
        setBoxHighlighted(7);
    }
}

```

```

private void box7TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable[7]) {
        clickable[7] = false;
        count = count + 1;
        setBoxColour(7, config[7]);
        if (control == 0) {
            control = config[7];
        } else {
            if (control == config[7]) {
                control = 0;
            }
        }

        playAudio(4);
        gameScore += Math.round((5 - (float) (count / 2)) * (((float) (msTimeMaximum -
msTimeInterval) / (float) (msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
        gameScore2L.setText("" + gameScore);

        if (gameScore > highScore) {
            if (newHighScore && highScore != 0) {
                playAudio(6);
                JOptionPane.showMessageDialog(this, "New High Score Acheived !!");
            }
        }
    }
}

```

```

        newHighScore = false;
    }

    highScore = gameScore;
    hScore2L.setText("" + highScore);

    try {
        Class.forName("java.sql.DriverManager");
        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
        Statement stmt = (Statement) con.createStatement();
        String query = "UPDATE highscores SET MatchTheColours = " + highScore + " WHERE
Username = " + username + " ";
        stmt.executeUpdate(query);
    } catch (Exception e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
}

    if (count == 8) {
        newLevel();
    }
    else {
        gameOver();
    }
}
}
}

```

### Text Field – 8:

```

private void box8TFMouseEntered(java.awt.event.MouseEvent evt) {
    if (clickable[8]) {
        setBoxHighlighted(8);
    }
}

private void box8TFMouseExited(java.awt.event.MouseEvent evt) {
    if (clickable[8]) {
        setBoxNormal(8);
    }
}

private void box8TFMousePressed(java.awt.event.MouseEvent evt) {
    if (clickable[8]) {
        setBoxPressed(8);
    }
}

private void box8TFMouseReleased(java.awt.event.MouseEvent evt) {
    if (clickable[8]) {
        setBoxHighlighted(8);
    }
}

```

```

private void box8TFMouseClicked(java.awt.event.MouseEvent evt) {
    if (clickable[8]) {
        clickable[8] = false;
        count = count + 1;
        setBoxColour(8, config[8]);
        if (control == 0) {
            control = config[8];
        } else {
            if (control == config[8]) {
                control = 0;

                playAudio(4);
                gameScore += Math.round((5 - (float) (count / 2)) * (((float) (msTimeMaximum -
msTimeInterval) / (float) (msTimeMaximum - msTimeMinimum)) * (maxMultiple - 1)) + 1));
                gameScore2L.setText("" + gameScore);

                if (gameScore > highScore) {
                    if (newHighScore && highScore != 0) {
                        playAudio(6);
                        JOptionPane.showMessageDialog(this, "New High Score Acheived !!");
                        newHighScore = false;
                    }

                    highScore = gameScore;
                    hScore2L.setText("" + highScore);

                    try {
                        Class.forName("java.sql.DriverManager");
                        Connection con = (Connection)
DriverManager.getConnection("jdbc:mysql://localhost:3306/gamezonedata", "root", "12345");
                        Statement stmt = (Statement) con.createStatement();
                        String query = "UPDATE highscores SET MatchTheColours = " + highScore + " WHERE
Username = " + username + ",";
                        stmt.executeUpdate(query);
                    } catch (Exception e) {
                        JOptionPane.showMessageDialog(this, e.getMessage());
                    }
                }

                if (count == 8) {
                    newLevel();
                }
            } else {
                gameOver();
            }
        }
    }
}

```



## Game Over:


### Button – Play Again:

```
private void playAgainBtnActionPerformed(java.awt.event.ActionEvent evt) {  
    gameOverPanel.setVisible(false);  
  
    exit = true;  
  
    control = 0;  
    count = 0;  
  
    msTimeInterval = msTimeMaximum;  
  
    gameScore = 0;  
    gameScore2L.setText("" + gameScore);  
  
    hScore2L.setText("" + highScore);  
  
    if (highScore == 0) {  
        newHighScore = false;  
    } else {  
        newHighScore = true;  
    }  
  
    setAllBoxesNormal();  
  
    correctMsgL.setVisible(false);  
    instructionsMsg1L.setVisible(false);  
    instructionsMsg2L.setVisible(false);  
  
    startBtn.setText("Start Game");  
    startBtn.setVisible(true);  
  
    startMsg1L.setVisible(true);  
    startMsg2L.setVisible(true);  
    startMsg3L.setVisible(true);  
  
    gamePanel.setVisible(true);  
    playAudio(1);  
}
```



End of Game

# MYSQL TABLE DESCRIPTION



```

MySQL 5.7 Command Line Client
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.7.12-log MySQL Community Server (GPL)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> USE gameZoneData
Database changed
mysql> DESCRIBE userAccounts;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Username | varchar(15) | NO | PRI | NULL | |
| Password | varchar(1000) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> _
  
```



```

MySQL 5.7 Command Line Client
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.7.12-log MySQL Community Server (GPL)

Copyright (c) 2000, 2016, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> USE gameZoneData;
Database changed
mysql> DESCRIBE highScores;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| Username | varchar(15) | YES | | NULL | |
| RememberTheSequence | int(11) | YES | | NULL | |
| GetTheSquares | int(11) | YES | | NULL | |
| MatchTheColours | int(11) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> _
  
```

# BIBLIOGRAPHY

1. NCERT – A Textbook on Informatics Practices for Class XII
2. Informatics Practices A Textbook for Class XI – Sumita Arora
3. <http://www.codeproject.com/>
4. <http://www.javatpoint.com/>
5. <http://www.dbforums.com/>
6. <http://www.tutorialspoint.com/>
7. <http://www.codejava.net/>
8. <http://forums.netbeans.org/>
9. <http://docs.oracle.com/javase/>
10. <http://alvinalexander.com/>
11. <http://stackoverflow.com/>
12. <http://www.java2s.com/>

# **CD CONTAINING THE PROGRAM**