Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

Java & Scala Laboratory

Mini Project Report

Treasure Hunt

A game of wit and luck

By

Farin Khan	**60009210140**
Durgesh Pandit	**60009210145**
Sanika Tawate	**60009210161**

Guide

Prof. Archana Nanade

Table of Contents

1. Introduction

"Treasure Hunt" project, a meticulously crafted Java Swing application that seamlessly integrates the Minesweeper game with advanced features and intricate rules. This project represents a convergence of algorithmic design, graphical user interface development, and strategic gaming elements.

The game unfolds across five progressively challenging levels of Minesweeper, characterized by expanding grid sizes and escalating mine counts. The intricate gameplay introduces lifelines and a clandestine wild card entry, strategically embedded to enhance the complexity of decision-making.

As users navigate through the Minesweeper levels, they encounter not only the challenges of minefield exploration but also the intricacies of diverse hint games—Tic Tac Toe, Whac-A-Mole, Simon Says, and Hangman—each requiring distinct algorithms and programming implementations.

This project incorporates a sophisticated scoring system, rewarding successful Minesweeper completions, treasure unlocks, and victories in the hint games. Through this fusion of technical prowess and gaming strategy, the "Treasure Hunt" project not only tests algorithmic efficiency but also challenges users to navigate the complexities of graphical user interface development in Java Swing.

2. Technology Used

Programming Language: Java (Enabling logic building)

User Interface Framework: Java Swing (Enabling interaction and game play)

Database connectivity: JDBC API (Enabling the leaderboard feature)

Database: MySQL (Enabling player and scores storage and query)

Click here to view the complete game rules.

3. Code Snippets

Main file - App.java:

```java
public class App {
    public static int i = 0;
    public static String hints[] = new String[5];
    public static int score = 0;
    public static void main(String[] args) throws Exception {
        int flag = 0;
        DatabaseManager db = new DatabaseManager();
        db.createDB();
        db.createTable();
        Register register = new Register();
        new Rules();
        Minesweeper minesweeper1 = new Minesweeper(1);
        minesweeper1.playGame();
        if (minesweeper1.win() == 1){
            score += 2;
            TicTacToe tictactoe = new TicTacToe();
            if (tictactoe.win() == 1){
                score += 2;
                hints[i] = "The first digit is greater than 5.";
                i++;
            }
            Minesweeper minesweeper2 = new Minesweeper(2);
            minesweeper2.playGame();
            if (minesweeper2.win() == 1){
                score += 4;
                SimonSays simonsays = new SimonSays();
                if(simonsays.win() == 1){
                    score += 2;
                    hints[i] = "The second digit is an odd number.";
                    i++;
                }
                Minesweeper minesweeper3 = new Minesweeper(3);
                minesweeper3.playGame();
                if (minesweeper3.win() == 1){
                    score += 8;
                    WhacAMole whacamole = new WhacAMole();
                    if(whacamole.win() == 1){
                        score += 2;
                        hints[i] = "The sum of all digits is 20.";
                        i++;
                    }
                    Minesweeper minesweeper4 = new Minesweeper(4);
                    minesweeper4.playGame();
                    if (minesweeper4.win() == 1){
                        score += 16;
                        DurgeshHangmanGUI hangman = new DurgeshHangmanGUI();
                        if(hangman.win() == 1){
                            score += 2;
                                hints[i] = "The last digit is one less than the
third digit.";
                            i++;
                        }
                        Minesweeper minesweeper5 = new Minesweeper(5);
                        minesweeper5.playGame();
                        if (minesweeper5.win() == 1){
                            score += 32;
                                          UnlockTreasure unlocktreasure = new
UnlockTreasure(hints);
                            unlocktreasure.showFrame();
                            if (unlocktreasure.win()==1){
                                score += 100;
                                flag = 1;
                            }
                        }
                    }
                }
            }
        }
        db.updateScores(register.name, score);
        new LeaderBoard(flag);
    }
}
```

Minesweeper.java

```java
import java.awt.*;
import java.awt.event.*;
import java.util.ArrayList;
import java.util.Random;
import java.util.concurrent.CountDownLatch;
import javax.swing.*;

public class Minesweeper {
    private CountDownLatch gameLatch = new CountDownLatch(1);
    private class MineTile extends JButton {
        int r;
        int c;
        public MineTile(int r, int c) {
            this.r = r;
            this.c = c;
        }
    }

    int boardWidth = 680;
    int boardHeight = 680;
    int numRows;
    int numCols;
    int mineCount;
    int tileSize;

    MineTile[][] board;
    MineTile treasure;
    ArrayList<MineTile> mineList;

    JFrame frame = new JFrame("Minesweeper");
    JLabel textLabel = new JLabel();
    JLabel life = new JLabel();
    JPanel textPanel = new JPanel();
    JPanel boardPanel = new JPanel();

    int flag = 0;
    int lifeCount = 3;
    Random random = new Random();

    int tilesClicked = 0;
    boolean gameOver = false;

    Minesweeper(int level) {
        numRows = 4*level;
        numCols = numRows;
        tileSize = (int) boardHeight/level;

        mineCount = numRows;
        board = new MineTile[numRows][numCols];

        frame.setSize(boardWidth, boardHeight);
        frame.setLocationRelativeTo(null);
        frame.setResizable(false);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setLayout(new BorderLayout());

        textLabel.setFont(new Font("Arial", Font.BOLD, 25));
        textLabel.setText("Minesweeper: " + Integer.toString(mineCount));

        life.setFont(new Font("Segoe UI Symbol", Font.BOLD, 25));
        life.setText("          Lifes:  \u2764  \u2764  \u2764 ");

        textPanel.add(textLabel, BorderLayout.CENTER);
        textPanel.add(life, BorderLayout.EAST);
        frame.add(textPanel, BorderLayout.NORTH);

        boardPanel.setLayout(new GridLayout(numRows, numCols));
        frame.add(boardPanel);
    }

    void playGame(){
        for (int r = 0; r < numRows; r++) {
            for (int c = 0; c < numCols; c++) {
                MineTile tile = new MineTile(r, c);
                board[r][c] = tile;

                tile.setFocusable(false);
                tile.setMargin(new Insets(0, 0, 0, 0));
                tile.setFont(new Font("Segoe UI Symbol", Font.PLAIN, 20));
                tile.addMouseListener(new MouseAdapter() {
                    @Override
                    public void mousePressed(MouseEvent e) {
                        if (gameOver) {
                            return;
                        }
                        MineTile tile = (MineTile) e.getSource();
                        if (e.getButton() == MouseEvent.BUTTON1) {
                            if (tile.getText() == "") {
                                if (mineList.contains(tile)) {
                                    if (lifeCount == 0)
                                        revealMines();
                                    else {
                                        lifeCount--;
                                        tile.setText("\u2620");
                                        if(lifeCount == 2)
                                            life.setText("             Lifes:
\u2764  \u2764 ");

                                        else if(lifeCount == 1)
                                            life.setText("
Lifes:  \u2764 ");

                                        else
                                            life.setText("            Lifes:
");

                                    }
                                }
                                else if(tile == treasure){
                                    gameOver = true;
                                    textLabel.setText("Success! Wild card
revealed.");

                                    flag = 1;
                                    int delay = 3000;
                                    Timer timer = new Timer(delay, new
ActionListener() {
                                        @Override
                                        public void
actionPerformed(ActionEvent e) {
                                            frame.dispose();
                                            gameLatch.countDown();
                                        }
                                    });

                                    timer.setRepeats(false);
                                    timer.start();
                                }
```

```java
                            else {
                                checkMine(tile.r, tile.c);
                            }
                        }
                    }
                    else if (e.getButton() == MouseEvent.BUTTON3) {
                        if (tile.getText() == "" && tile.isEnabled()) {
                            tile.setText("\u2691");
                        }
                        else if (tile.getText() == "\u2691") {
                            tile.setText("");
                        }
                    }
                }
            });
            boardPanel.add(tile);
        }
    }
    frame.setVisible(true);

    setMines();
    setTreasure();
    try {
        gameLatch.await();
    } catch (InterruptedException err) {
        err.printStackTrace();
    }
}

void setTreasure() {
    while(true){
        int r = random.nextInt(numRows);
        int c = random.nextInt(numCols);
        if(mineList.contains(board[r][c]))
            continue;
        else{
            treasure = board[r][c];
            break;
        }
    }
}
```

```java
    void setMines() {
        mineList = new ArrayList<MineTile>();
        int mineLeft = mineCount;
        while (mineLeft > 0) {
            int r = random.nextInt(numRows);
            int c = random.nextInt(numCols);

            MineTile tile = board[r][c];
            if (!mineList.contains(tile)) {
                mineList.add(tile);
                mineLeft -= 1;
            }
        }
    }

    void revealMines() {
        for (int i = 0; i < mineList.size(); i++) {
            MineTile tile = mineList.get(i);
            tile.setText("\u2620");
        }

        gameOver = true;
        textLabel.setText("Game Over!");
        life.setText("                Lifes:    ");
        int delay = 3000;
        Timer timer = new Timer(delay, new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                frame.dispose();
                gameLatch.countDown();
            }
        });

        timer.setRepeats(false);
        timer.start();
    }

    void checkMine(int r, int c) {
        if (r < 0 || r >= numRows || c < 0 || c >= numCols) {
            return;
        }
        MineTile tile = board[r][c];
        if (!tile.isEnabled()) {
            return;
        }
        tile.setEnabled(false);
        tilesClicked += 1;

        int minesFound = 0;

        minesFound += countMine(r-1, c-1);
        minesFound += countMine(r-1, c);
        minesFound += countMine(r-1, c+1);

        minesFound += countMine(r, c-1);
        minesFound += countMine(r, c+1);

        minesFound += countMine(r+1, c-1);
        minesFound += countMine(r+1, c);
        minesFound += countMine(r+1, c+1);

        if (minesFound > 0) {
            tile.setText(Integer.toString(minesFound));
        }
        else {
            tile.setText("");

            checkMine(r-1, c-1);
            checkMine(r-1, c);
            checkMine(r-1, c+1);

            checkMine(r, c-1);
            checkMine(r, c+1);

            checkMine(r+1, c-1);
            checkMine(r+1, c);
            checkMine(r+1, c+1);
        }

        if (tilesClicked == numRows * numCols - mineList.size()) {
            gameOver = true;
            textLabel.setText("Mines Cleared!");
            flag = 1;
            int delay = 3000;
            Timer timer = new Timer(delay, new ActionListener() {
                @Override
                public void actionPerformed(ActionEvent e) {
                    frame.dispose();
                    gameLatch.countDown();
                }
            });

            timer.setRepeats(false);
            timer.start();
        }
    }

    int countMine(int r, int c) {
        if (r < 0 || r >= numRows || c < 0 || c >= numCols) {
            return 0;
        }
        if (mineList.contains(board[r][c])) {
            return 1;
        }
        return 0;
    }

    int win(){
        if (flag == 1){
            return(1);
        }
        return(0);
    }
}
```

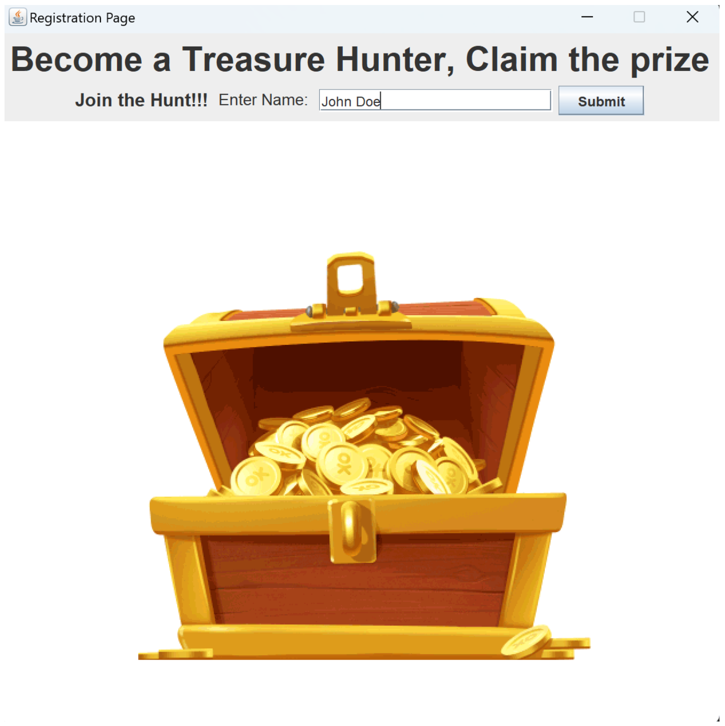
DatabaseManager.java

```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

public class DatabaseManager {
    String url = "jdbc:mysql://localhost:3306/";
    String databaseName = "db";
    String username = "root";
    String password = "password";
    String name;
    Connection connection;
    Statement statement;

    DatabaseManager(){
        try {
            Class.forName("com.mysql.cj.jdbc.Driver");
            this.connection = DriverManager.getConnection(url+databaseName,
username, password);
            this.statement = connection.createStatement();
            System.out.println("Database Connected Successfully");
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void closeConnection() {
        try {
            if (statement != null) {
                statement.close();
            }
            if (connection != null) {
                connection.close();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    public void createDB(){
        try{
            String query = "CREATE DATABASE IF NOT EXISTS "+ databaseName;
            statement.execute(query);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void createTable(){
        try {
            String query = "CREATE TABLE IF NOT EXISTS scores (name
VARCHAR(20), score INT(4))";
            statement.execute(query);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public void createUser(String name){
        try {
            String query = "INSERT INTO scores (name, score) VALUES ('" + name
+ "', 0)";
            statement.execute(query);
            System.out.println("Data Inserted Successfully");
        } catch (Exception e) {
            System.out.println("Retry Inserting Data");
        }
    }

    public void updateScores(String name, int score){
        try{
              String query = "UPDATE scores SET score=? WHERE name=?";
              PreparedStatement ps = connection.prepareStatement(query);

              ps.setInt(1, score);
              ps.setString(2, name);

              int affectedRows = ps.executeUpdate();
              if(affectedRows<=0){
                System.out.println("User not found: " + name);
              } else {
                System.out.println("Score updated successfully for " + name);
              }
        } catch (Exception e) {
            System.out.println("Retry Updating");
        }
    }

    public ResultSet readData(){
        try {
            String query = "Select * FROM scores ORDER BY score DESC";
            ResultSet rs = statement.executeQuery(query);
            return rs;
        } catch (Exception e) {
            System.out.println("Error Reading Data");
            return null;
        }
    }

    public void deleteUser(String name){
        try {
        String query = "DELETE FROM scores WHERE name=?";
        PreparedStatement ps = connection.prepareStatement(query);
        ps.setString(1, name);
        int affectedRows = ps.executeUpdate();
        if (affectedRows > 0) {
            System.out.println("Deleted entry for " + name);
        } else {
            System.out.println("User not found: " + name);
        }
        } catch (Exception e) {
            System.out.println("Error deleting entry for " + name);
        }
    }
}
```

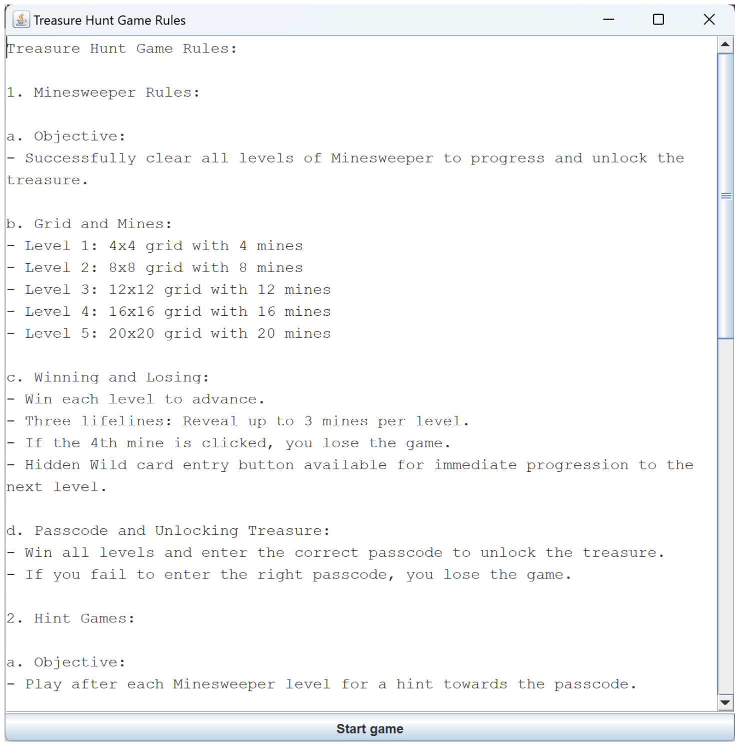
Click here to view the entire project code (including hint games' & other windows')

4. Results

Login page:

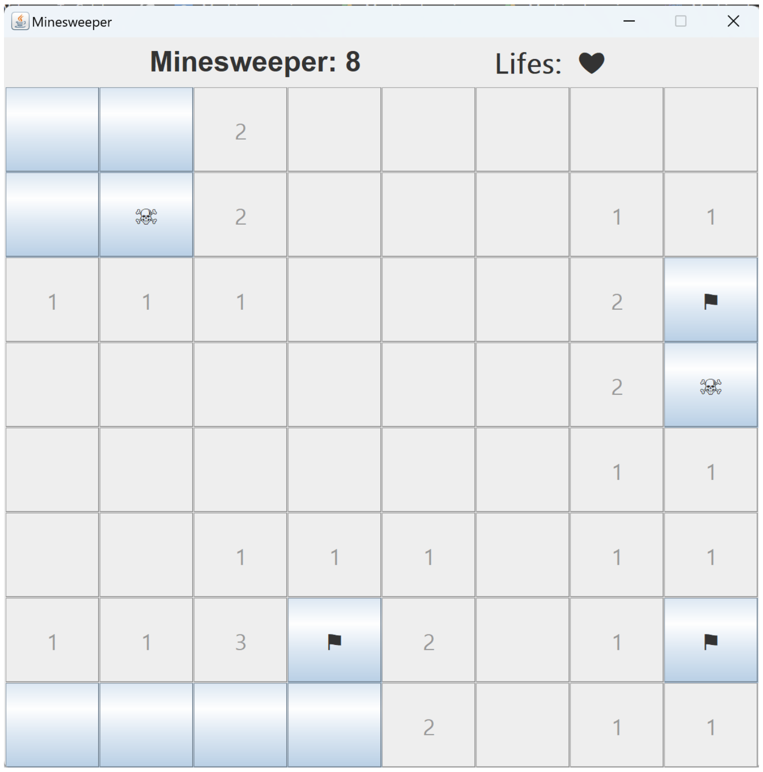

Registration Page
Become a Treasure Hunter, Claim the prize
Join the Hunt!!! Enter Name:
John Doe
Submit

Rules:

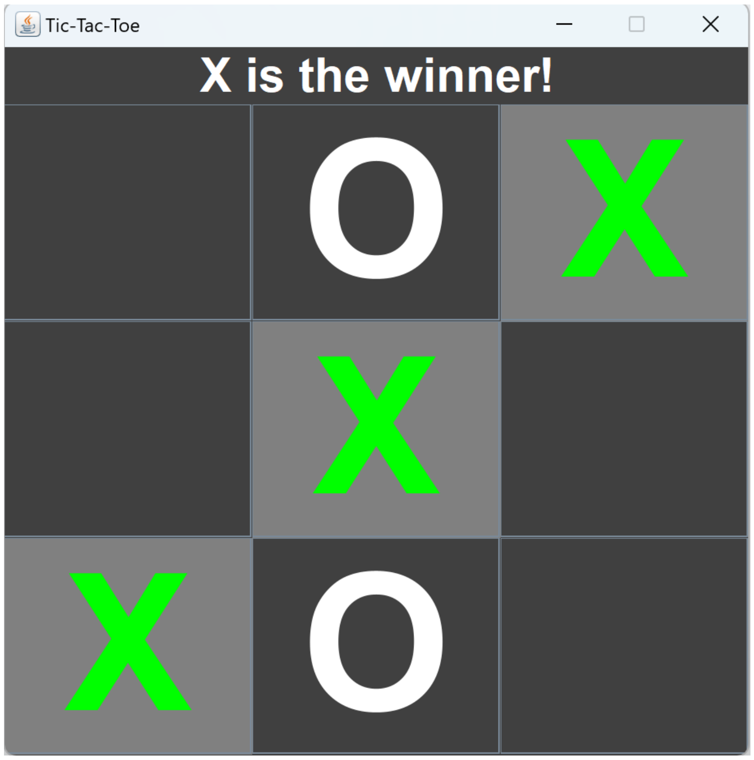


```
Treasure Hunt Game Rules:

1. Minesweeper Rules:

a. Objective:
- Successfully clear all levels of Minesweeper to progress and unlock the
treasure.

b. Grid and Mines:
- Level 1: 4x4 grid with 4 mines
- Level 2: 8x8 grid with 8 mines
- Level 3: 12x12 grid with 12 mines
- Level 4: 16x16 grid with 16 mines
- Level 5: 20x20 grid with 20 mines

c. Winning and Losing:
- Win each level to advance.
- Three lifelines: Reveal up to 3 mines per level.
- If the 4th mine is clicked, you lose the game.
- Hidden Wild card entry button available for immediate progression to the
next level.

d. Passcode and Unlocking Treasure:
- Win all levels and enter the correct passcode to unlock the treasure.
- If you fail to enter the right passcode, you lose the game.

2. Hint Games:

a. Objective:
- Play after each Minesweeper level for a hint towards the passcode.
```

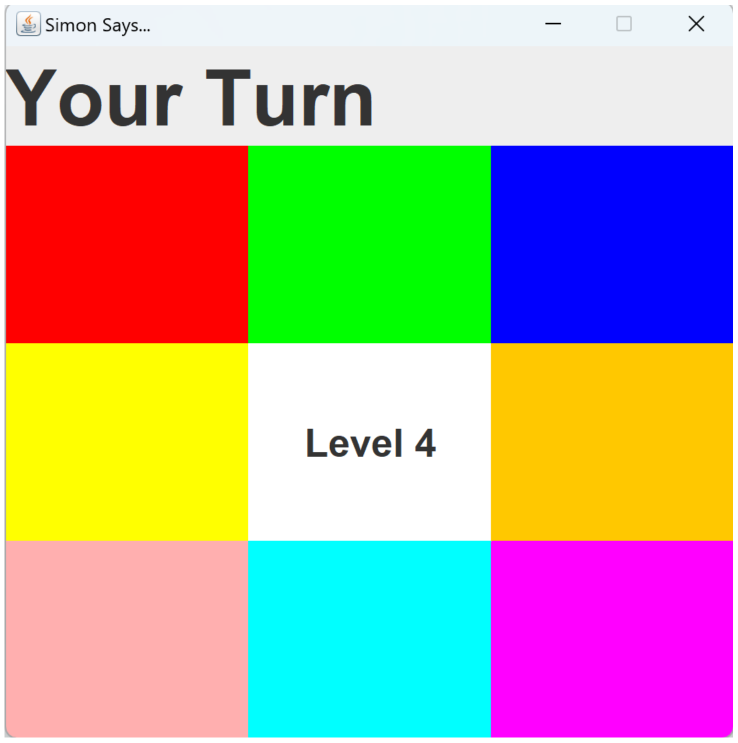
Minesweeper:



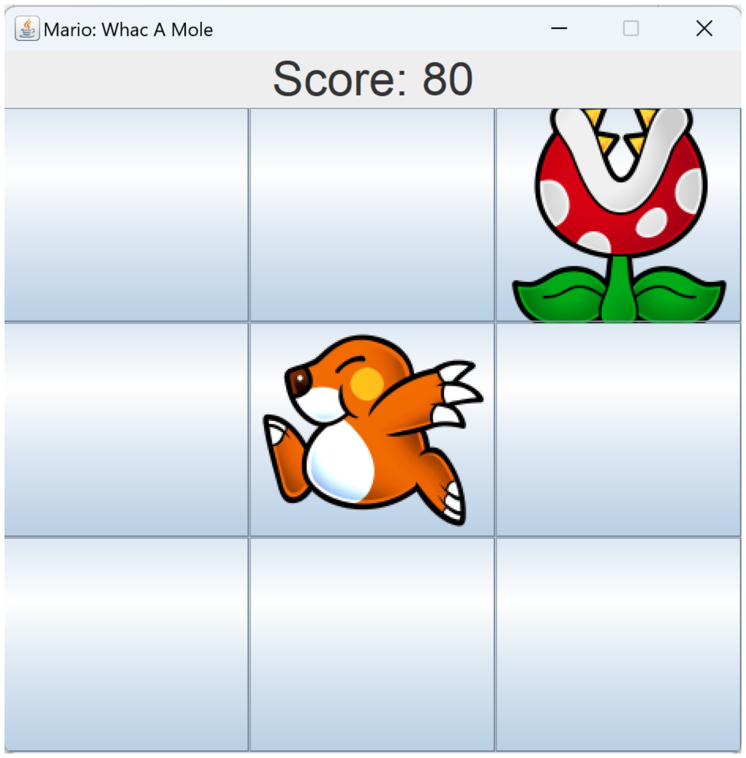
Tic Tac Toe:



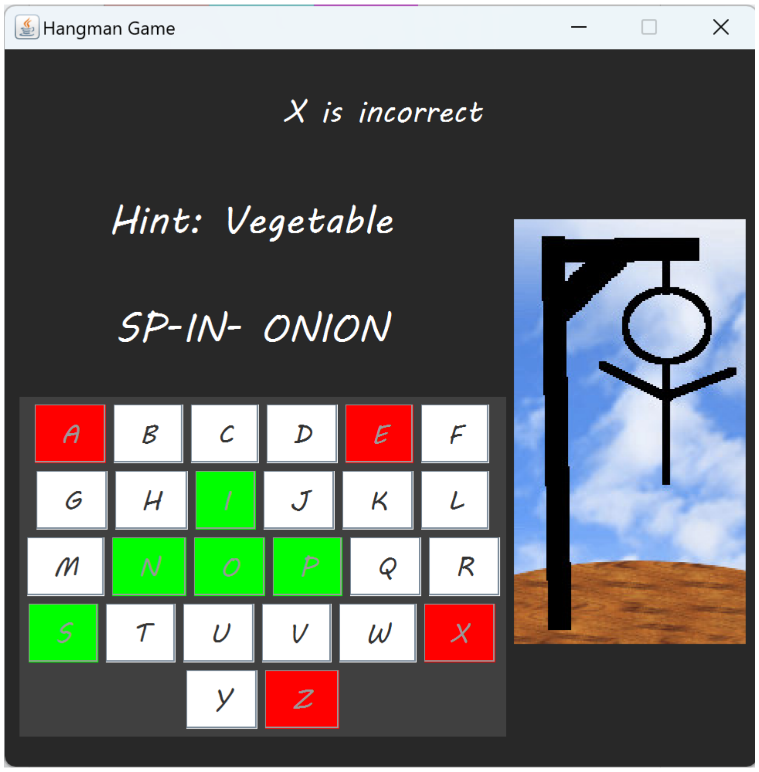
Simon Says:



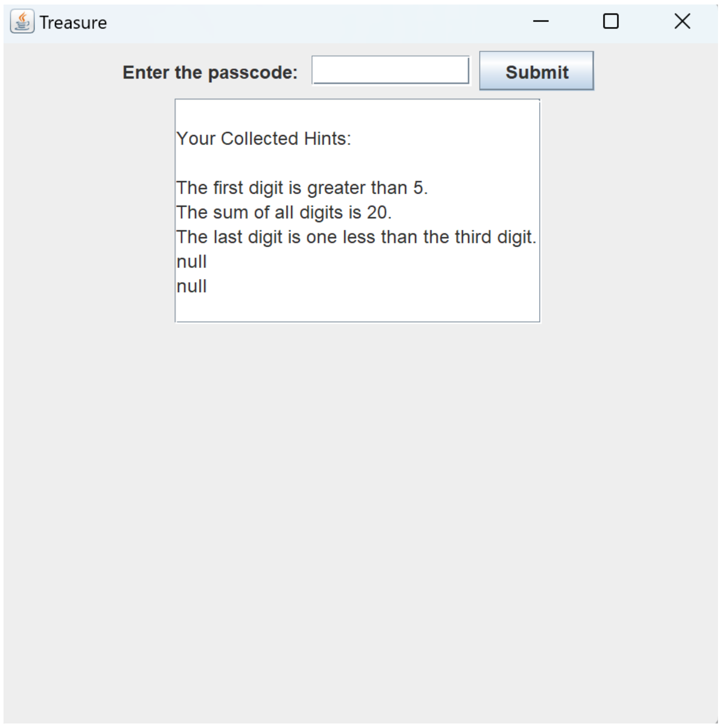
Whac A Mole:



Hangman:

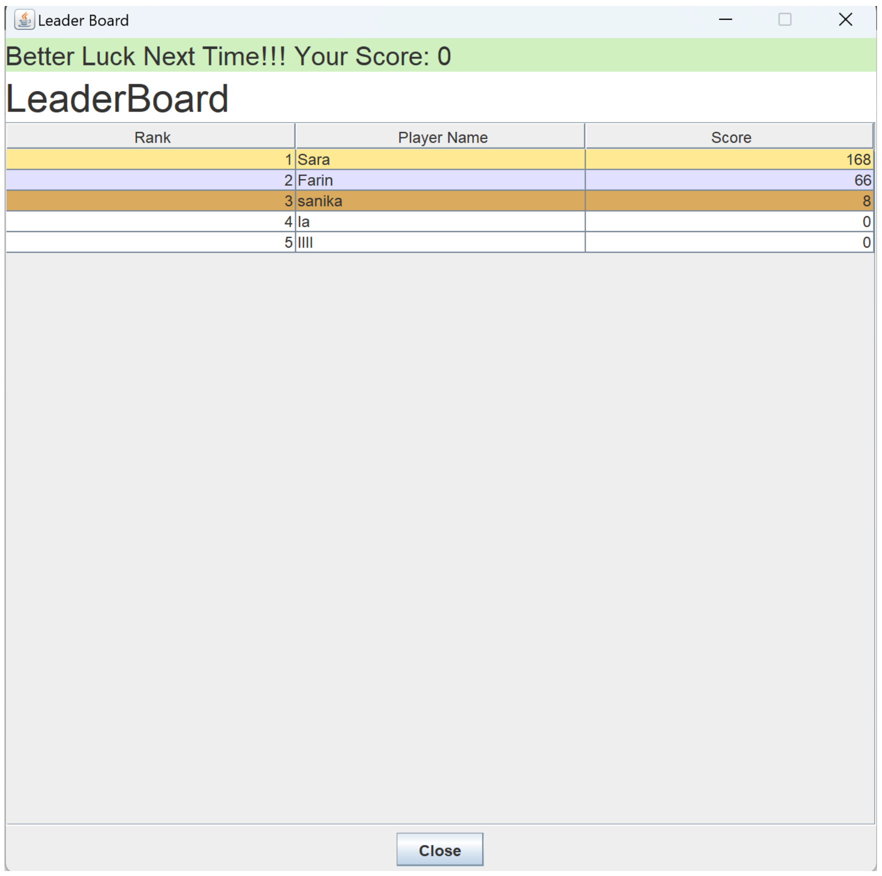


Unlock Treasure:



```
Your Collected Hints:

The first digit is greater than 5.
The sum of all digits is 20.
The last digit is one less than the third digit.
null
null
```

Result with Leaderboard:



Better Luck Next Time!!! Your Score: 0

LeaderBoard

Rank	Player Name	Score
1	Sara	168
2	Farin	66
3	sanika	8
4	la	0
5	llll	0