PROJECT HANGMAN

Rheeya Uppaal

TABLE OF CONTENTS

1. Introduction

1.1 Overview

Interactive entertainment, as is popularly known, is the order of the day across the globe and encompasses dozens of job disciplines and employs thousands of people worldwide. The contemporary world of information technology has ushered a wonderful surge into the gaming industry of a wide variety and profile of games that not only provide entertainment value, but also give impetus to young minds on the educational platform. Such games are played and relished by all age brackets since it stimulates the mind positively and towards achievement of a purpose of growth and evolution of the mind.

Keeping the same trend in mind, this project has incorporated a game called **'Hangman'** using MySQL and Java Net Beans. This game tests the language and vocabulary skills of the player over multiple words. It encourages the player to think of a word while the system suggests letters randomly placed, to fill the blanks provided.

The word to guess is represented by a row of dashes, giving the number of letters and category of the word. If the player suggests a letter which occurs in the word, the system accepts it and writes it in all its correct positions. If the suggested letter does not occur in the word, the system draws one element of the hanged man stick figure as a tally mark. A tab button also offers clues and hints for the player to judge the word as he plays his turn. The game is over when the stick figure is complete and ready to be hung.

1.2 Purpose

The game is extremely interesting as it poses a challenge to one's vocabulary skills. This project contains about 150 words that may be guessed by a player.

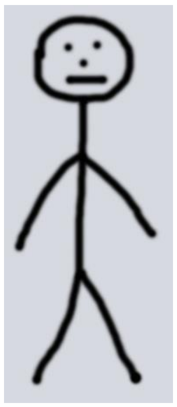
The Strategy to play this game - In the English language, the 12 most commonly occurring letters are, in descending order: <u>e-t-a-o-i-n-s-h-r-d-l-u</u>. This and other letter-frequency lists are used by the guessing player to increase the odds when it is their turn to guess.

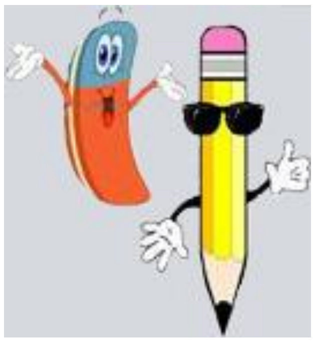
1.3 Scope of the Project

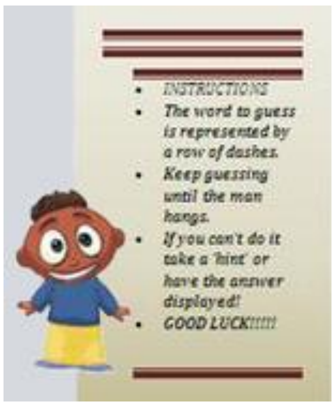
The field of software engineering and development is fast growing, especially the sector of game development which is growing even more rapidly.

In a game project such as this, the product is the game. However, a game is much more than software. It has to provide content to become enjoyable; and the quality for that cannot be measured. The software part of the project is not the only one; it must be considered in connection to all other parts: The environment of the game and so on.

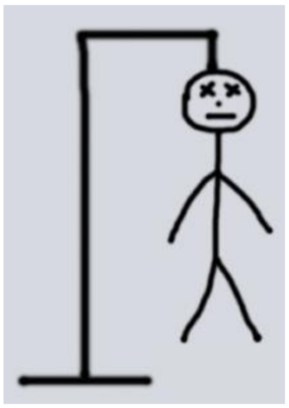
Since a large majority of the world indulges in playing games, the scope of games in general is huge. Hangman, being a simple paper game, is even more widespread and well known, increasing the scope of this project dramatically.







- *INSTRUCTIONS*
- *The word to guess is represented by a row of dashes.*
- *Keep guessing until the man hangs.*
- *If you can't do it take a 'hint' or have the answer displayed!*
- *GOOD LUCK!!!!!!*

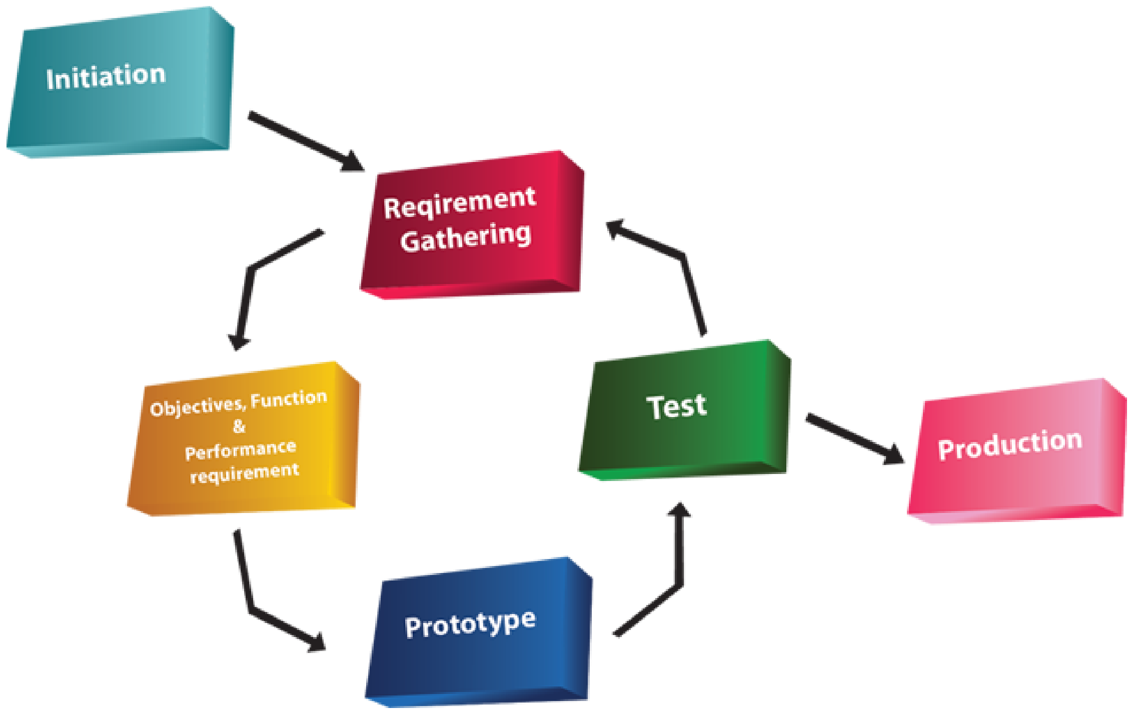


2. Overall Description

2.1 Process Model

The “Prototyping” software process model has been adopted for this project. The model is a systems development method (SDM) in which a prototype (an early approximation of a final system or product) is built, tested, and then reworked as necessary until an acceptable prototype is finally achieved.

The basic idea is that throwaway prototype is built to understand the requirements. This prototype is developed based on the currently known requirements. Development of the prototype undergoes design, coding and testing. These phases are not performed very formally or thoroughly. By using this prototype, the client can get an "actual feel" of the system, since the interactions with prototype can enable the client to better understand the requirements of the desired system.



The Prototyping Process Model

This model works best in scenarios where not all of the project requirements are known in detail ahead of time. It is an iterative, trial-and-error process that takes place between the developers and the users.

There are several steps in the Prototyping Model:

1. The new system requirements are defined in as much detail as possible
2. A preliminary design is created for the new system.
3. A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
4. The users thoroughly evaluate the first prototype, noting its strengths and weaknesses, what needs to be added, and what should to be removed. The developer collects and analyzes the remarks from the users.
5. The first prototype is modified, based on the comments supplied by the users, and a second prototype of the new system is constructed.
6. The second prototype is evaluated in the same manner as was the first prototype.
7. The preceding steps are iterated as many times as necessary, until the users are satisfied that the prototype represents the final product desired.
8. The final system is constructed, based on the final prototype.
9. The final system is thoroughly evaluated and tested. Routine maintenance is carried out on a continuing basis to prevent large-scale failures and to minimize downtime.

Advantages of Prototyping:

1. Users are actively involved in the development
2. It provides a better system to users, as users have natural tendency to change their mind in specifying requirements and this method of developing systems supports this user tendency.
3. Since in this methodology a working model of the system is provided, the users get a better understanding of the system being developed.
4. Errors can be detected much earlier as the system is made side by side.
5. Quicker user feedback is available leading to better solutions.

Disadvantages:

1. Leads to implementing and then repairing way of building systems.
2. Practically, this methodology may increase the complexity of the system as scope of the system may expand beyond original plans.

2.2 Functional Requirements

Normal requirements consist of the objectives and goals that are stated during the meeting with the relevant customers. Normal requirements of our project are:

1. User friendly efficient and lucrative system
2. Minimum maintenance cost
3. Availability of expected requirements within the required configuration
4. Ease of operation
5. Measured coding

Expected requirements are implicit to the system and may be so fundamental that the customer does not state them. Their absence is a cause for dissatisfaction:

1. Development system with limited cost
2. Minimum hardware requirements relevant to the game
3. Efficient design of the entire system

The game also has the following, important requirements:

1. User Interfaces – The presence of a menu and other elements to make a game user friendly and easy to understand are important.
2. Hardware Interfaces – A software should ideally be platform independent and should not require any specific software to run.

2.3 Non-functional Requirements

Nonfunctional requirements deal with the characteristics of the system which cannot be expressed as functions of the system, portability of the system, usability of the system, etc.

Nonfunctional requirements may include:

- Reliability issues
- Accuracy of results
- Human –Computer interface issues
- Constraints on the system implementation, etc.

3 Project Design

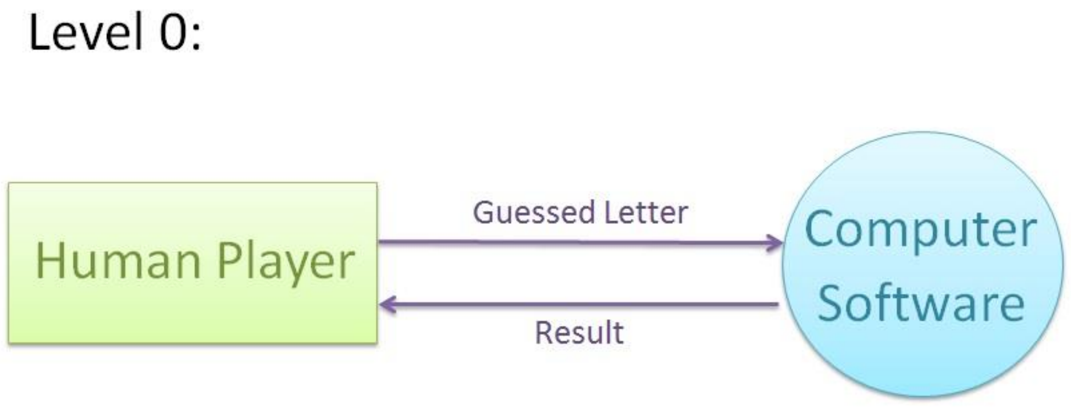
3.1 Analysis Models

3.1.1 Functional Requirements (DFD)

A data flow diagram is a graphical representation that depicts information flow and the transforms that are applied as data moves from input to output. Process specification (PSPEC) is used to specify the procedure details.

A DFD will look at from where data comes, where it goes and where it is stored. However it does not store information about processing timing and sequence.

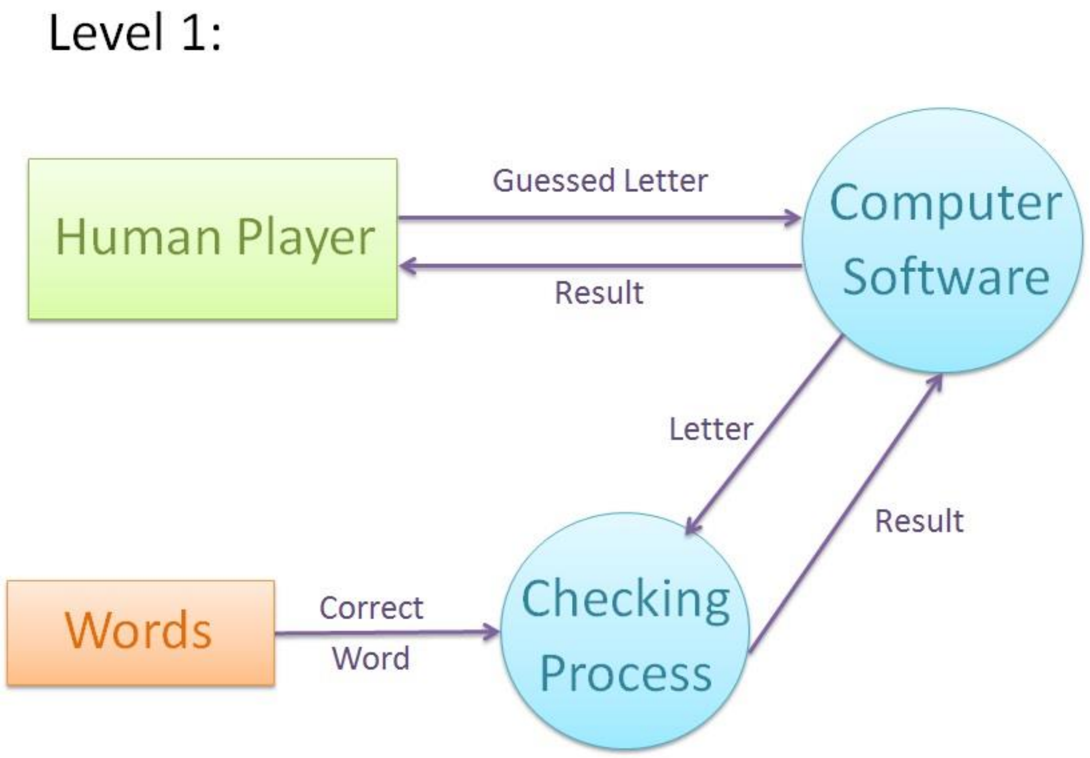
DFDs should not be confused with flowcharts. Flowcharts show the flow of control while DFDs show the flow of data. DFDs do not store control elements.



Data Flow Diagram – Level 0

The data flow diagram for this project consists of two major components: The human player and the Computer Software. The player sends a letter as input. In return, the computer software displays the result, showing whether the guess is correct.

The functions have been described in more detail in the level one diagram.



Data Flow Diagram – Level 1

3.1.2 Entity Relationship Diagram

The data model of a project consists of three interrelated pieces of information:

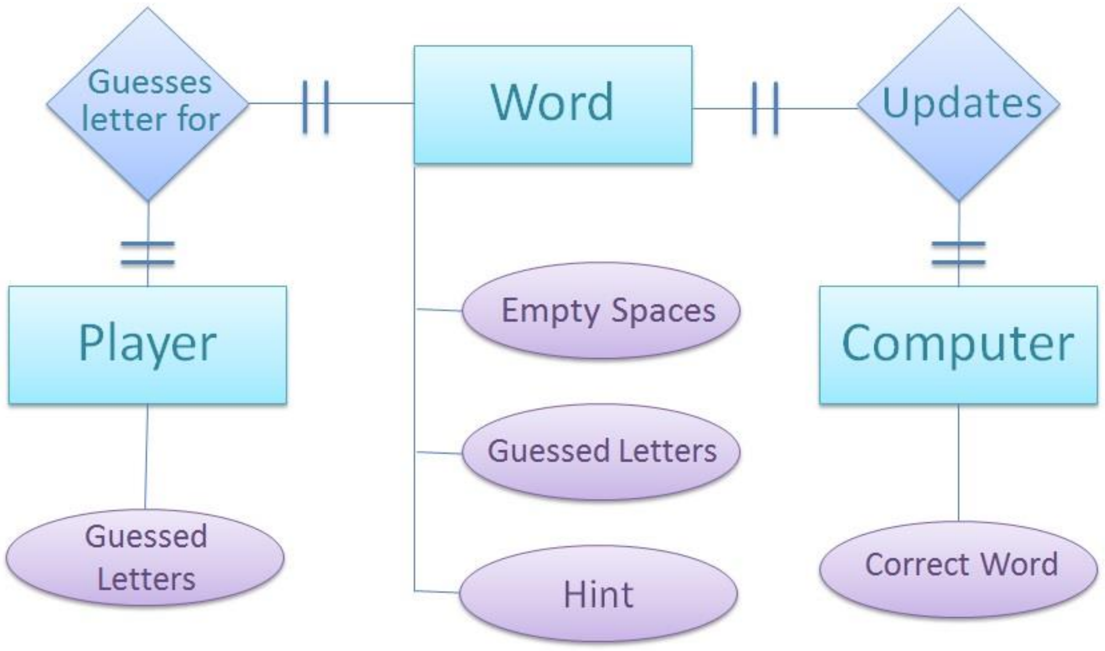
- The data object (Entity)
- The attributes that describe the data object
- The relationships that connect data objects to one another

The data object description (DOD) incorporates the data objects and all of their attributes. The Entiry Relationship Diagram (ERD) enables a software engineer to identify data objects and their relationships using a graphical notation.

A data object can be an external entity (E.g. anything that produces or consumes information), a thing (e.g., a report or a display), an occurrence (e.g., a telephone call) or event (e.g., an alarm), a role (e.g., salesperson), an organizational unit (e.g., accounting department), a place (e.g., a warehouse), or a structure (e.g., a file). Data objects are related to one another.

Attributes define the properties of a data object. They can be used to name an instance of the data object, describe the instance or make reference to another instance in another table. For example, a car is defined in terms of make, model, ID number, body type, color and owner. The body of the table represents specific instances of the data object.

The relationships are always defined by the context of the problem that is being analyzed. A relationship denotes a specific connection between the entities. Relationship pairs are bidirectional.

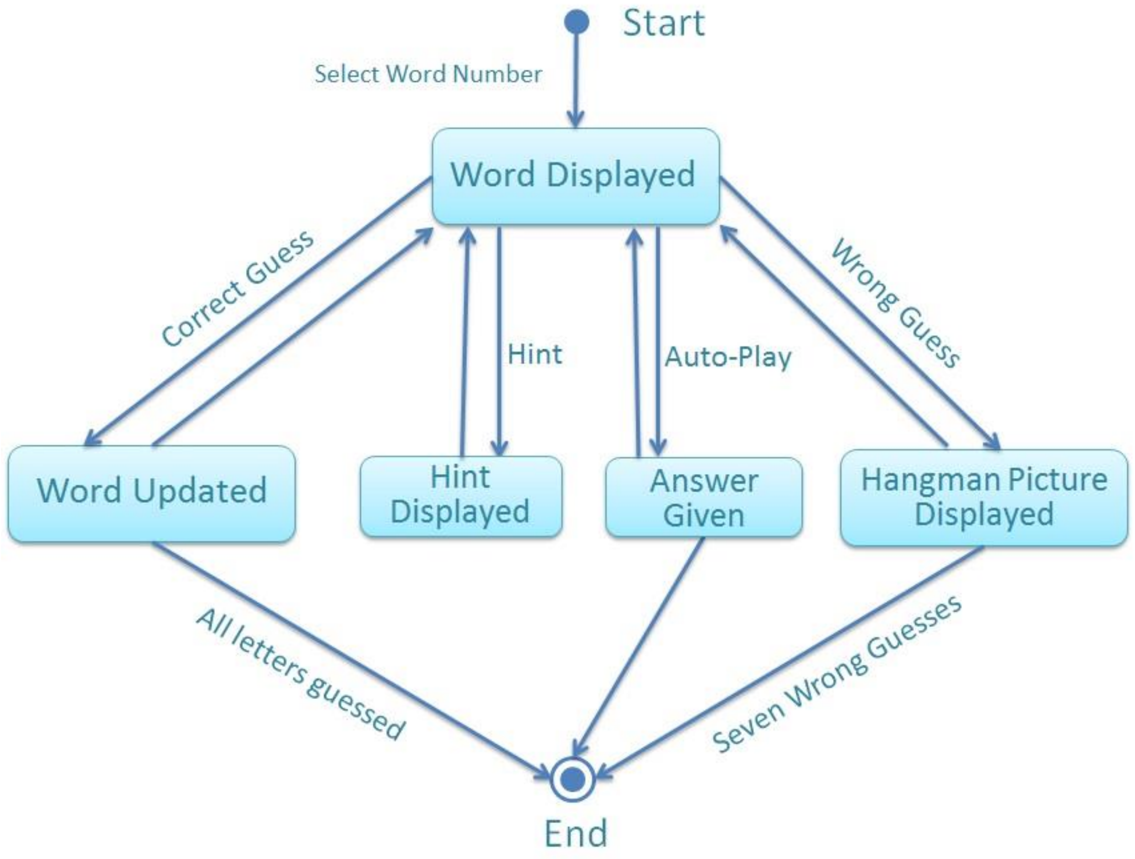


Entity – Relationship Diagram

3.1.3 State Transition Diagram

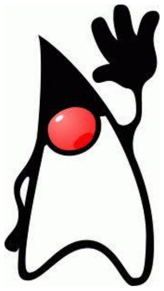
Behavioral modeling is an operational principle for all requirement analysis methods. Control information is contained in control specification or CSPEC.

The state transition diagram (STD) represents the behavior of a system by depicting its states and the events that cause the system to change state. The STD indicates what actions are taken as a consequence of a particular event. A state is any observable mode of behavior.



State Transition Diagram

In this, the game begins with the user selecting a number to select a game. A word, is chosen accordingly. Now, if the user makes a correct guess, the guess is updated to the word and displayed. If the user asks for a hint, the hint is displayed, following which the game resumes. If the user selects the auto-play option, the answer is displayed and the game ends. If the user makes an incorrect guess, a hangman picture is displayed, indicating a wrong guess. If seven wrong guesses have been made, the final hangman picture is displayed and the game ends.



THE JAVA SOURCE CODE BEHIND

HANGMAN

Start Page - Frame

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class StartPage extends JFrame implements ActionListener
{
    JButton newgame, highscores, exit;
    JLabel icon;

    StartPage()
    {
        newgame=new JButton("New Game");
        highscores=new JButton("High Scores");
        exit=new JButton("Exit");

        icon=new JLabel("");
        icon.setIcon(new ImageIcon("C:/Users/Rheeya/Desktop/"
        + "Second Year/Java and Web Technologies/Project/"
                + "Hangman/Pictures/Hangman.jpg"));

        newgame.addActionListener(this);
        highscores.addActionListener(this);
        exit.addActionListener(this);
    }

    public void actionPerformed(ActionEvent e)
    {
        String s=e.getActionCommand();

        if(s.equals("New Game"))
            {SelectGameType sgt= new SelectGameType();}

        if(s.equals("High Scores"))
            {HighScores hs=new HighScores();}

        if(s.equals("Exit"))
            {System.exit(0);}
    }

    public void prepareGUI()
    {
        setVisible(true);
        setSize(400,350);
        setTitle("Hangman");

        setLayout(new GridBagLayout());
        GridBagConstraints g=new GridBagConstraints();

        g.gridy=0;          add(icon,g);
        g.gridy=1;          add(new JLabel(" "),g);
        g.gridy=2;          add(newgame,g);
        g.gridy=3;          add(new JLabel(" "),g);
        g.gridy=4;          add(highscores,g);
        g.gridy=5;          add(new JLabel(" "),g);
        g.gridy=6;          add(exit,g);
    }

    public static void main(String args[])
    {
        StartPage s=new StartPage();
        s.prepareGUI();
    }
}
```

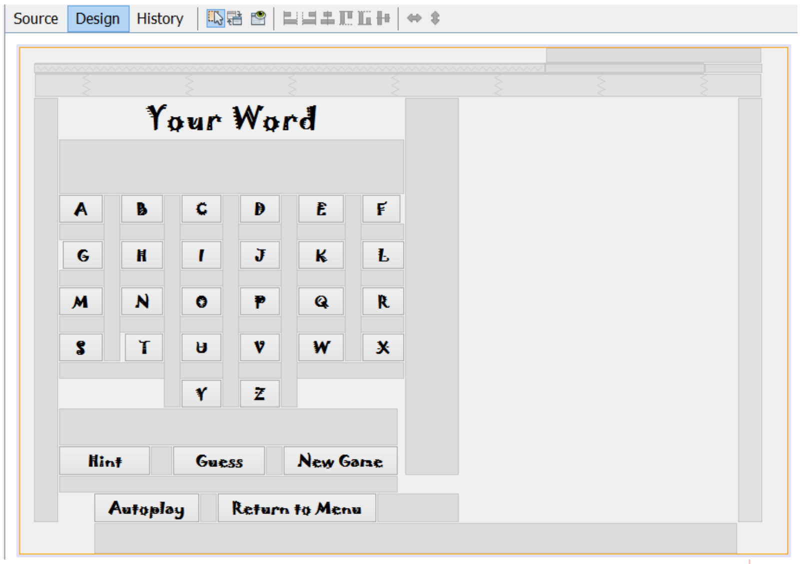
Select Game Type - Frame

```java
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class SelectGameType extends JFrame implements ActionListener
{
    JButton words, movies, exit;
    SelectGameType()
    {
        words=new JButton("Words");
        movies=new JButton("Movies");
        exit=new JButton("Return to Main Menu");

        words.addActionListener(this);
        movies.addActionListener(this);
        exit.addActionListener(this);

        setSize(400,350);
        setVisible(true);
        setLayout(new GridBagLayout());
        GridBagConstraints g=new GridBagConstraints();

        JLabel icon = new JLabel("");
        icon.setIcon(new ImageIcon("C:/Users/Rheeya/Desktop"
                + "/Second Year/Java and Web Technologies/"
                + "Project/IP Project - Hangman/Hangman/"
                + "Pictures/Final/Hangman.jpg"));

        g.gridy=0;          add(icon,g);
        g.gridy=1;          add(new JLabel("  "),g);
        g.gridy=2;          add(words,g);
        g.gridy=3;          add(new JLabel("  "),g);
        g.gridy=4;          add(movies,g);
        g.gridy=5;          add(new JLabel("  "),g);
        g.gridy=6;          add(new JLabel(""),g);
        g.gridy=7;          add(exit,g);
    }

    public void actionPerformed(ActionEvent e)
    {
        String s=e.getActionCommand();

        if(s.equals("Words"))
            {Hangman h=new Hangman("Words");}

        if(s.equals("Movies"))
            {Hangman h=new Hangman("Movies");}

        if(s.equals("Return to Main Menu"))
            this.dispose();
    }
}
```

High Scores – Frame

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
import java.sql.*;
import javax.swing.table.*;

public class HighScores extends JFrame implements ActionListener
{
    JButton exit;
    JTable t;
    JLabel l1, l2, l3;
    HighScores()
    {
        t=new JTable(0,2);
        l1=new JLabel("Name");
        l2=new JLabel("Score");
        l3=new JLabel("HIGH SCORES");
        exit=new JButton("Return to Main Menu");
        exit.addActionListener(this);

        setSize(100,200);
        setVisible(true);
        setTitle("Hangman");
        setLayout(new GridBagLayout());
        GridBagConstraints g=new GridBagConstraints();

        g.gridx=0;  g.gridy=0;  g.gridwidth=2;  add(l3,g);
        g.gridx=0;  g.gridy=1;  g.gridwidth=1;  add(new JLabel(""),g);
        g.gridx=0;  g.gridy=2;  g.gridwidth=1;  add(l1,g);
        g.gridx=1;  g.gridy=2;  g.gridwidth=1;  add(l2,g);
        g.gridx=0;  g.gridy=3;  g.gridwidth=1;  add(new JLabel(""),g);
        g.gridx=0;  g.gridy=4;  g.gridwidth=2;  add(t,g);
        g.gridx=0;  g.gridy=5;  g.gridwidth=1;  add(new JLabel(""),g);
        g.gridx=0;  g.gridy=6;  g.gridwidth=2;  add(exit,g);

        //Connectivity Statements
        try
        {
            DefaultTableModel obj=(DefaultTableModel)t.getModel();
            Class.forName("java.sql.DriverManager");
            Connection con=(Connection)DriverManager.getConnection("jdbc:mysql://localhost:3306/Hangman","root","MySQL");
            Statement st=(Statement)con.createStatement();
            String query= "select Name, Score from HighScores order by Score desc;";
            ResultSet rs=st.executeQuery(query);

            while(rs.next())
            {
                String name=rs.getString("Name");
                String score=rs.getString("Score");
                obj.addRow(new Object[]{name,score});
            }
            rs.close();
            st.close();
            con.close();
        }
        catch(Exception e)
            {JOptionPane.showMessageDialog(null,"Error");}
    }

    public void actionPerformed(ActionEvent e)
    {this.dispose();}
}
```

Hangman – Frame



```
import javax.swing.JOptionPane;
import javax.swing.ImageIcon;
import java.sql.*;
import com.mysql.jdbc.Connection;
import com.mysql.jdbc.Statement;

public class Hangman extends javax.swing.JFrame {
    public Hangman(String option) {
        setVisible(true);
        setTitle("Hangman");
        gametype=option;
        initComponents();
    }

static String gametype;
static char letter; //letter that is guessed by the user
static int n=0, score=10; //n = No of blanks in a word
static String hw="", cw="", meaning="", jw="";
static int correct=0, wrong=0; //No. of correct and incorrect guesses by the user in one game

@SuppressWarnings("unchecked")
    Generated Code

private void QuitActionPerformed(java.awt.event.ActionEvent evt) {
this.dispose();
}

private void New_GameActionPerformed(java.awt.event.ActionEvent evt) {
    A.setEnabled(false);
    B.setEnabled(true);
    C.setEnabled(true);
    D.setEnabled(true);
    E.setEnabled(false);
    F.setEnabled(true);
    G.setEnabled(true);
    H.setEnabled(true);
    I.setEnabled(false);
    J.setEnabled(true);
    K.setEnabled(true);
    L.setEnabled(true);
    M.setEnabled(true);
    N.setEnabled(true);
    O.setEnabled(false);
    P.setEnabled(true);
    Q.setEnabled(true);
    R.setEnabled(true);
    S.setEnabled(true);
    T.setEnabled(true);
    U.setEnabled(false);
    V.setEnabled(true);
    W.setEnabled(true);
    X.setEnabled(true);
    Y.setEnabled(true);
    Z.setEnabled(true);
    Hint.setEnabled(true);
    Guess.setEnabled(true);
    Autoplay.setEnabled(true);

    Word.setText("Your Word");
    Pic.setIcon(new ImageIcon(""));
    wrong=0;
    correct=0;
    n=0;
    score=10;

    int co=0, gn;
    //Input Game number
    String GN="";
    if(gametype.equals("Words"))
        GN=JOptionPane.showInputDialog(this,"Please enter a Game Number between 1 and 150");
    if(gametype.equals("Movies"))
        GN=JOptionPane.showInputDialog(this,"Please enter a Game Number between 1 and 10");
    gn=Integer.parseInt(GN);

/*Connectivity Statements*/
try
{
    Class.forName("java.sql.DriverManager");
    Connection con=(Connection)DriverManager.getConnection("jdbc:mysql://localhost:3306/Hangman","root","MySQL");
    Statement st=(Statement)con.createStatement();

    //Select game type
    String query="";
    if(gametype.equals("Words"))
        query="select * from Words where No="+gn+";";
    if(gametype.equals("Movies"))
        query="select * from Movies where No="+gn+";";
    ResultSet rs=st.executeQuery(query);

    //Extract corresponding gn, word and hword from sql
    while(rs.next())
    {
        cw=rs.getString("Word");
        hw=rs.getString("HWord");
        meaning=rs.getString("Meaning");
        jw=rs.getString("Jword");
    }

    rs.close();
    st.close();
    con.close();
}
catch(Exception e)
    {JOptionPane.showMessageDialog(null,"Error");}
    Word.setText(hw);
    for(int i=0; i<hw.length();i++)
    {
        if(hw.charAt(i)=='_')
                n++;
    }
}

private void BActionPerformed(java.awt.event.ActionEvent evt) {
letter='B';
B.setEnabled(false);
}
private void CActionPerformed(java.awt.event.ActionEvent evt) {
letter='C';
C.setEnabled(false);
}
private void DActionPerformed(java.awt.event.ActionEvent evt) {
letter='D';
D.setEnabled(false);
}
private void EActionPerformed(java.awt.event.ActionEvent evt) {
letter='E';
E.setEnabled(false);
}
private void FActionPerformed(java.awt.event.ActionEvent evt) {
letter='F';
F.setEnabled(false);
}
private void GActionPerformed(java.awt.event.ActionEvent evt) {
letter='G';
G.setEnabled(false);
}
private void HActionPerformed(java.awt.event.ActionEvent evt) {
letter='H';
H.setEnabled(false);
}
private void IActionPerformed(java.awt.event.ActionEvent evt) {
letter='I';
I.setEnabled(false);
}
private void JActionPerformed(java.awt.event.ActionEvent evt) {
letter='J';
J.setEnabled(false);
}
private void KActionPerformed(java.awt.event.ActionEvent evt) {
letter='K';
K.setEnabled(false);
}
private void LActionPerformed(java.awt.event.ActionEvent evt) {
letter='L';
L.setEnabled(false);
}
private void MActionPerformed(java.awt.event.ActionEvent evt) {
letter='M';
M.setEnabled(false);
}
private void NActionPerformed(java.awt.event.ActionEvent evt) {
letter='N';
N.setEnabled(false);
}
private void OActionPerformed(java.awt.event.ActionEvent evt) {
letter='O';
O.setEnabled(false);
}
private void PActionPerformed(java.awt.event.ActionEvent evt) {
letter='P';
P.setEnabled(false);
}
private void QActionPerformed(java.awt.event.ActionEvent evt) {
letter='Q';
Q.setEnabled(false);
}
private void RActionPerformed(java.awt.event.ActionEvent evt) {
letter='R';
R.setEnabled(false);
}
private void SActionPerformed(java.awt.event.ActionEvent evt) {
letter='S';
S.setEnabled(false);
}
```

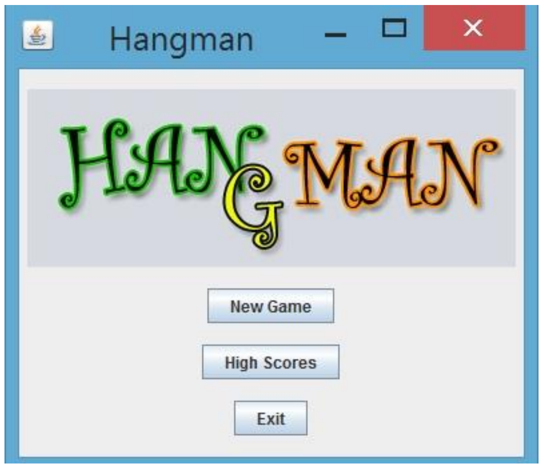
```
private void TActionPerformed(java.awt.event.ActionEvent evt) {
letter='T';
T.setEnabled(false);
}
private void UActionPerformed(java.awt.event.ActionEvent evt) {
letter='U';
U.setEnabled(false);
}
private void VActionPerformed(java.awt.event.ActionEvent evt) {
letter='V';
V.setEnabled(false);
}
private void WActionPerformed(java.awt.event.ActionEvent evt) {
letter='W';
W.setEnabled(false);
}
private void XActionPerformed(java.awt.event.ActionEvent evt) {
letter='X';
X.setEnabled(false);
}
private void YActionPerformed(java.awt.event.ActionEvent evt) {
letter='Y';
Y.setEnabled(false);
}
private void ZActionPerformed(java.awt.event.ActionEvent evt) {
letter='Z';
Z.setEnabled(false);
}
```

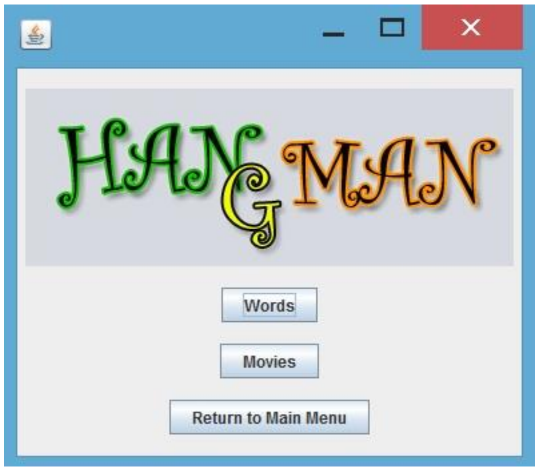
```
private void AutoplayActionPerformed(java.awt.event.ActionEvent evt) {
JOptionPane.showMessageDialog(null,"The Correct word is "+cw+". \n Meaning: "+meaning+".");
Hint.setEnabled(false);
Guess.setEnabled(false);
Autoplay.setEnabled(false);
}

private void HintActionPerformed(java.awt.event.ActionEvent evt) {
JOptionPane.showMessageDialog(null,""+meaning);
}

    private void GuessActionPerformed(java.awt.event.ActionEvent evt) {
    char g=letter;
    int len=jw.length(), i, j, co=0;
    for(i=0; i<len; i+=2)
    {
        char c=jw.charAt(i);
        if(g==c) //Replace
        {
            String s1, s2;
            s1=hw.substring(0,i);
            s2=hw.substring(i+1,len);
            hw=s1+c+s2;
            correct++;
            co++;
            Word.setText(hw);
        }
    }
    if(co==0)
    {
        wrong++;
        score--;
    }

    //Won Game
    int cor=0;
    for(i=0;i<hw.length();i++)
        if(hw.charAt(i)=='_')
            cor++;
    if(correct==n) //You Win!
    {
        JOptionPane.showMessageDialog(null,"Congratulations! "
                + "You Win! Your score is "+score+"/10!");

        //Check if a High Score is possible
        try
        {
            Class.forName("java.sql.DriverManager");
            java.sql.Connection con=(java.sql.Connection)DriverManager.
                 getConnection("jdbc:mysql://localhost:3306/Hangman",
                         "root","MySQL");
            java.sql.Statement st=(java.sql.Statement)con.createStatement();

            //Receive existing scores
            String q1= "select * from HighScores;";
            ResultSet rs=st.executeQuery(q1);

            //Compare to see if score is greater than others
            int hs=0;    //Flag bit
            while(rs.next())
            {
                int s=Integer.parseInt(rs.getString("Score"));
                if(score>s)
                    hs=1;
            }

            if(hs==1)    //High Score
            {
                String n=JOptionPane.showInputDialog(null,"High Score! Name:");
                String q2="insert into HighScores values('"+n+"',"+score+",null);";
                String q3="delete from HighScores where No=5;";
                st.executeUpdate(q2);        //Add to high score table
                st.executeUpdate(q3);        //Delete lowest high score
            }

            st.close();
            con.close();
        }
        catch(Exception e)
        {
            //JOptionPane.showMessageDialog(null,"Error");
            System.out.println(e.getMessage());
        }

        Hint.setEnabled(false);
        Autoplay.setEnabled(false);
        Guess.setEnabled(false);
        B.setEnabled(false);
        C.setEnabled(false);
        D.setEnabled(false);
        F.setEnabled(false);
        G.setEnabled(false);
        H.setEnabled(false);
        J.setEnabled(false);
        K.setEnabled(false);
        L.setEnabled(false);
        M.setEnabled(false);
        N.setEnabled(false);
        P.setEnabled(false);
        Q.setEnabled(false);
        R.setEnabled(false);
        S.setEnabled(false);
        T.setEnabled(false);
        V.setEnabled(false);
        W.setEnabled(false);
        X.setEnabled(false);
        Y.setEnabled(false);
        Z.setEnabled(false);
    }
    else
        if(co==0) //If after on
        {
            switch(wrong)
            {
               case 1: Pic.setIcon(new ImageIcon("1.jpg"));
                       break;
               case 2: Pic.setIcon(new ImageIcon("2.jpg"));
                       break;
               case 3: Pic.setIcon(new ImageIcon("3.jpg"));
                       break;
               case 4: Pic.setIcon(new ImageIcon("4.jpg"));
                       break;
               case 5: Pic.setIcon(new ImageIcon("5.jpg"));
                       break;
               case 6: Pic.setIcon(new ImageIcon("6.jpg"));
                       break;
            case 7: Pic.setIcon(new ImageIcon("7.jpg"));
                    JOptionPane.showMessageDialog(null,"Game Over");
                    Word.setText(jw);
                    Hint.setEnabled(false);
                    Autoplay.setEnabled(false);
                    Guess.setEnabled(false);
                    B.setEnabled(false);
                    C.setEnabled(false);
                    D.setEnabled(false);
                    F.setEnabled(false);
                    G.setEnabled(false);
                    H.setEnabled(false);
                    J.setEnabled(false);
                    K.setEnabled(false);
                    L.setEnabled(false);
                    M.setEnabled(false);
                    N.setEnabled(false);
                    P.setEnabled(false);
                    Q.setEnabled(false);
                    R.setEnabled(false);
                    S.setEnabled(false);
                    T.setEnabled(false);
                    V.setEnabled(false);
                    W.setEnabled(false);
                    X.setEnabled(false);
                    Y.setEnabled(false);
                    Z.setEnabled(false);
                    //You lose
        }
    }
}

public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    java.awt.EventQueue.invokeLater(new Runnable() {

        public void run() {
            new Hangman(gametype).setVisible(true);
        }
    });
}
// Variables declaration - do not modify
private javax.swing.JButton A;
private javax.swing.JButton Autoplay;
private javax.swing.JButton B;
private javax.swing.JButton C;
private javax.swing.JButton D;
private javax.swing.JButton E;
private javax.swing.JButton F;
private javax.swing.JButton G;
private javax.swing.JButton Guess;
private javax.swing.JButton H;
private javax.swing.JLabel Hangman;
private javax.swing.JButton Hint;
private javax.swing.JButton I;
private javax.swing.JButton J;
private javax.swing.JButton K;
private javax.swing.JButton L;
private javax.swing.JButton M;
private javax.swing.JButton N;
private javax.swing.JButton New_Game;
private javax.swing.JButton O;
private javax.swing.JButton P;
private javax.swing.JLabel Pic;
private javax.swing.JButton Q;
```

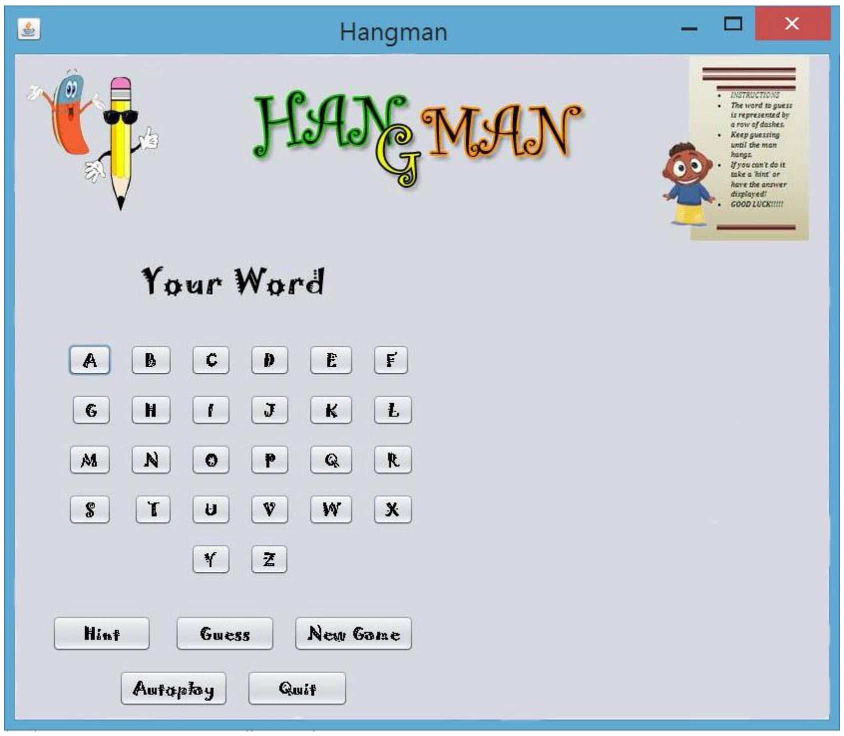
```
    private javax.swing.JButton Q;
    private javax.swing.JButton Quit;
    private javax.swing.JButton R;
    private javax.swing.JButton S;
    private javax.swing.JButton T;
    private javax.swing.JButton U;
    private javax.swing.JButton V;
    private javax.swing.JButton W;
    private javax.swing.JLabel Word;
    private javax.swing.JButton X;
    private javax.swing.JButton Y;
    private javax.swing.JButton Z;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    // End of variables declaration
}
```

THE OUTPUT FOR

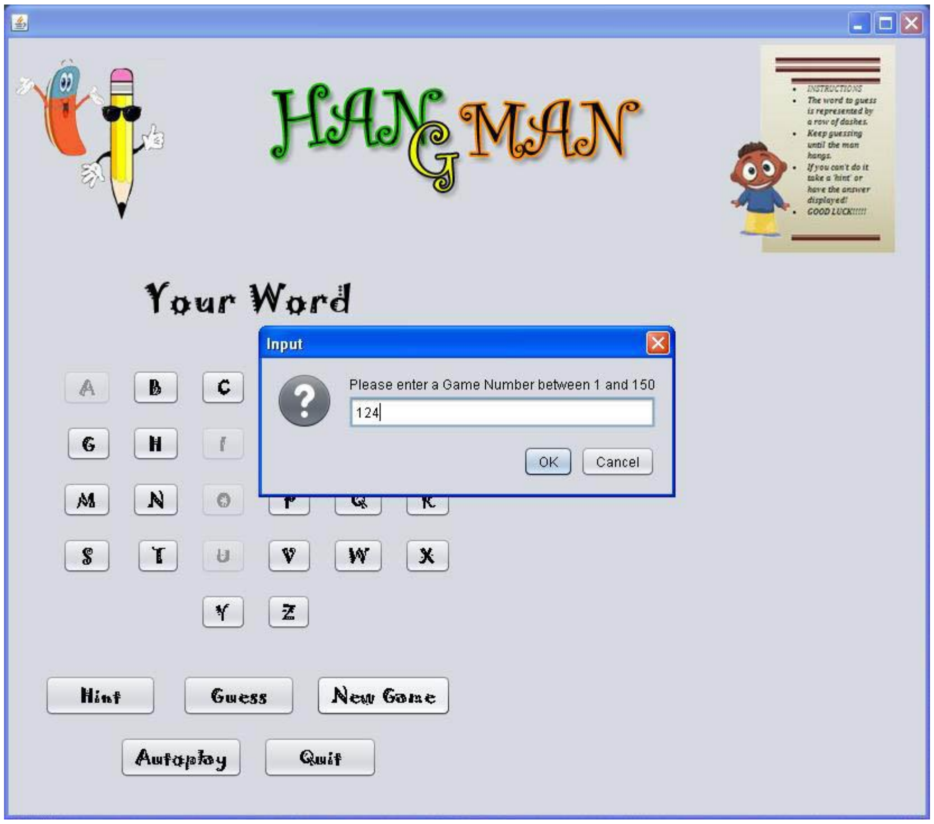
HANGMAN







- *INSTRUCTIONS*
- *The word to guess is represented by a row of dashes.*
- *Keep guessing until the man hangs.*
- *If you can't do it take a 'hint' or have the answer displayed!*
- *GOOD LUCK!!!!!!*

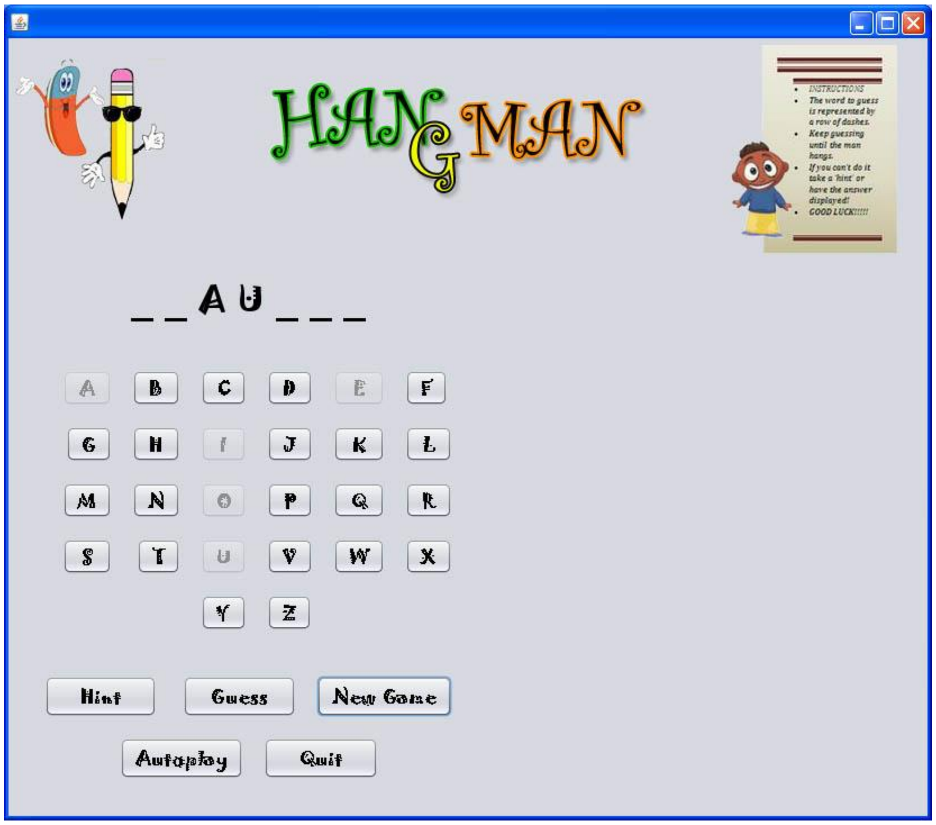


- INSTRUCTIONS
- The word to guess is represented by a row of dashes.
- Keep guessing until the man hangs.
- If you can't do it take a 'hint' or have the answer displayed!
- GOOD LUCK!!!!!

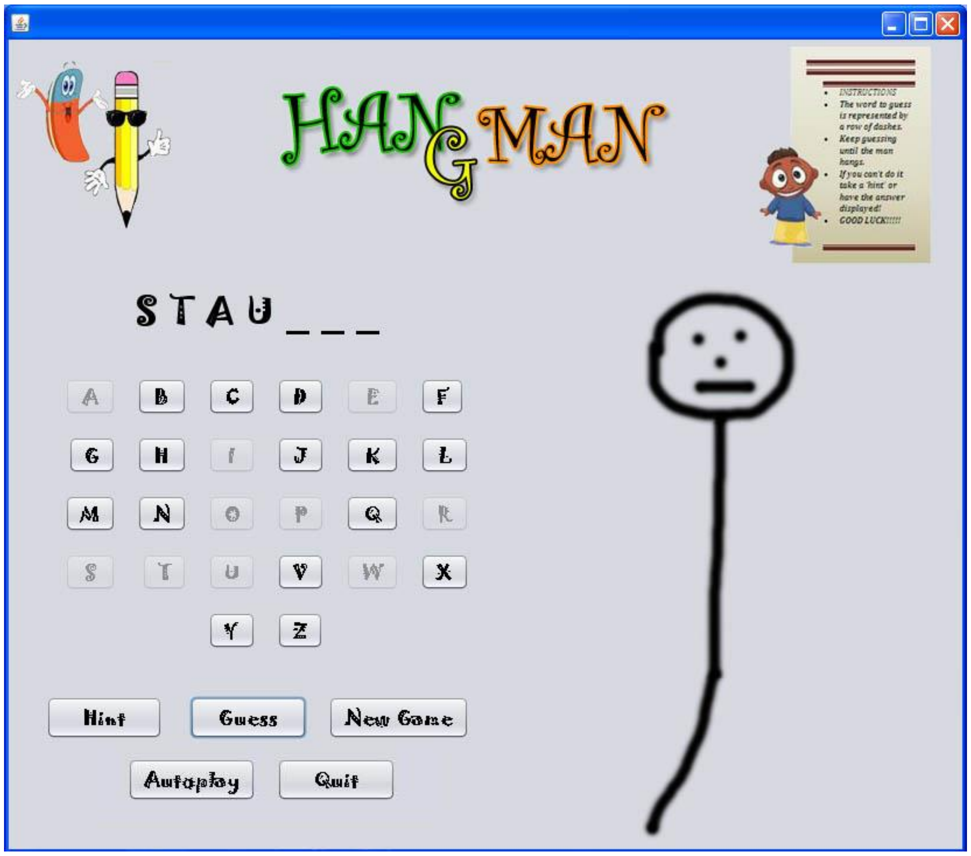
Input

Please enter a Game Number between 1 and 150

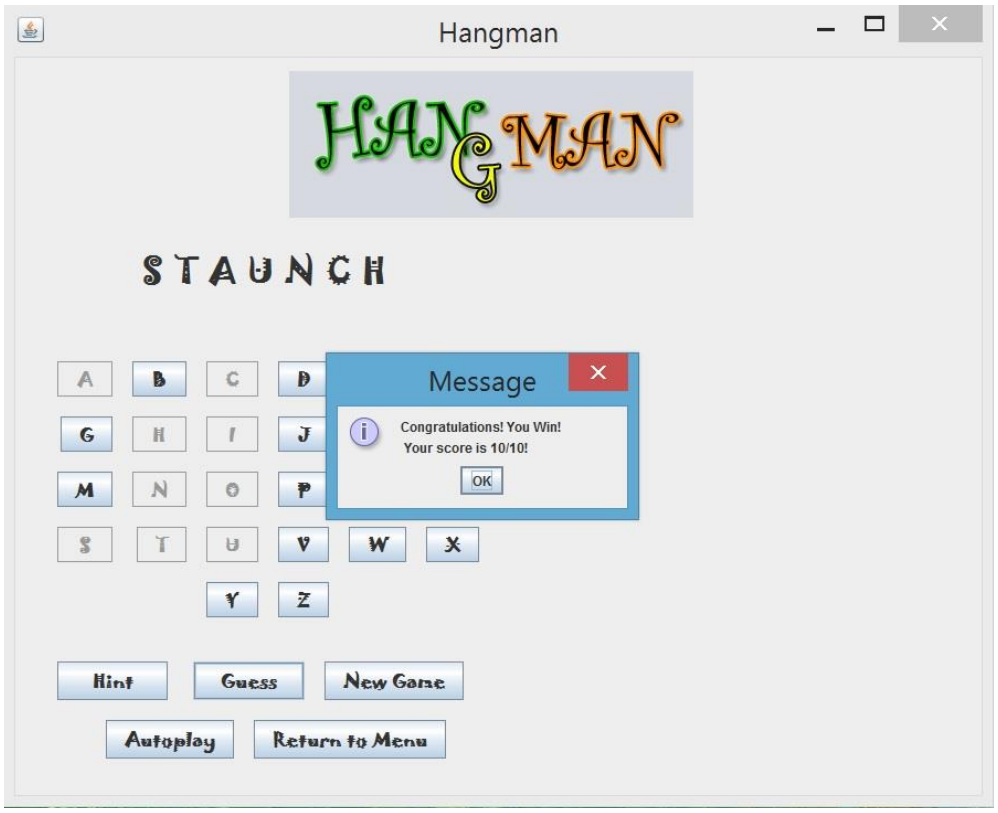
124



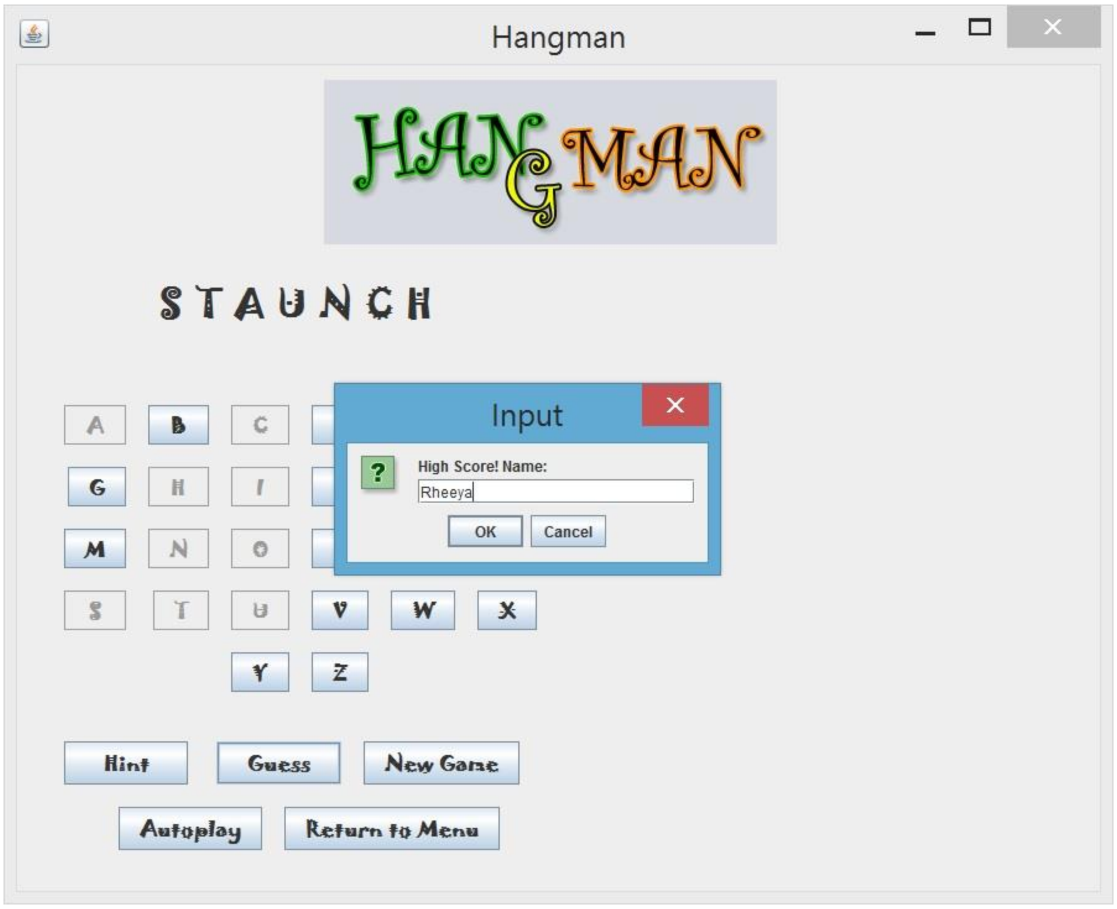
- INSTRUCTIONS
- The word to guess is represented by a row of dashes.
- Keep guessing until the man hangs.
- If you can't do it take a 'hint' or have the answer displayed!
- GOOD LUCK!!!!!



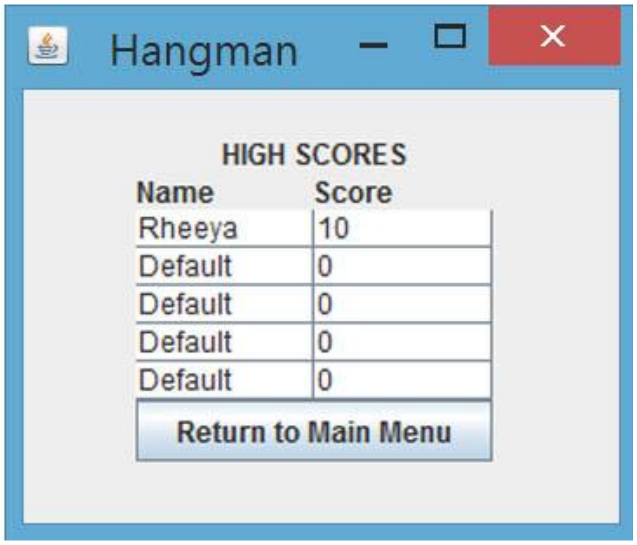
- INSTRUCTIONS
- The word to guess is represented by a row of dashes.
- Keep guessing until the man hangs.
- If you can't do it take a 'hint' or have the answer displayed!
- GOOD LUCK!!!!!



Congratulations! You Win!
Your score is 10/10!



High Score! Name:
Rheeya



HIGH SCORES

Name	Score
Rheeya	10
Default	0
Default	0
Default	0
Default	0