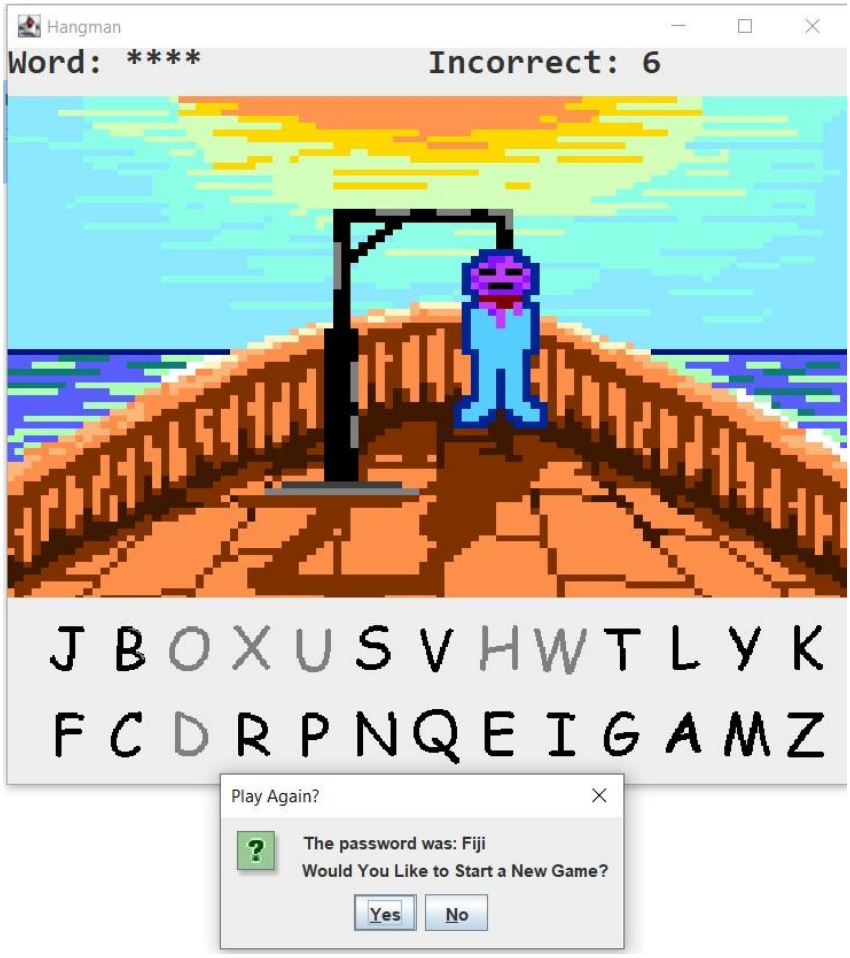
hangmanFinal

```
public class hangmanFinal {
    public static void main(String[] args){
        gameSquare newInstance = new gameSquare();
    }
}
```



hangman

```
public hangman() public hangman(String imageBaseName, String
    imageDirectory,
String imageType) private BufferedImage
    loadImage(String image_path) public void
    nextImage(int imageNo) public void losingScreen()
    public void winningScreen()
    private void loadNewImage(String suffix)
    @Override protected void
    paintComponent(Graphics newG)
```

tilesLetter

```
public tilesLetter() public tilesLetter(char imageLetter, String
    imageDIR, String
imageType) private BufferedImage loadImage(String
    image_path) private void loadNewImage(String
    suffix) private MouseListener tilesListener;
    public void addTileListener(MouseListener newOne)
    public void removeTileListener() public char
    guess() protected void paintComponent(Graphics
    newG)
```

letterRectangle

public letterRectangle()
public letterRectangle(String imgPassword, String imageDirectory, String imageType) private void loadRack() private void buildRack() public void attachListeners(MouseListener l) public void removeListeners()

gameSquare

```
 public gameSquare() private
    void initialize()
//Initialize method contains code that you would usually just see in a
//constructor, it has the basis of each instance of the game.
//But what if you wanted to start a new game without closing the window?
//This is where this method call comes in handy
//This has all the aspects of the game that WILL need to change
//like number of incorrect, status of Hangman, the letters
//but leaves out the stuff that doesn't need to change like
//the size of the window
//changes the number of Incorrect to 0
//changes the correct variable to a new jLabel “Word:”
//which is later appended with the word replaced by asterisks

    private void addCloseWindowListener() private void

    addTextPanel() private void addLetterRack() private void

    addHangman() public static String getCategory(String

    fileName) private void getPassword()

//This is what creates the first window that opens
//A string named options is made, it has all the options on the starting
//menu like “Lets Play” and “Quit”

//We make a JPanel that will include a JLabel and a JTextfield
//A label is made that will be next to the JTextfield where the user enters
//the word to be guessed. The Label is “Enter your word to be Guessed”.

    private void passwordGen()

    private void lengthCheck()
    private void newGameDialog()
    private class TileListener
    implements MouseListener
//important
//make a variable of the location of the mouse click, where it was clicked
on the screen
Object source = e.getSource();
```

```
//If the the mouseclick(source) it on of of the letters(letter
//objects of the tilesLetter class) then run this if statement
if(source instanceof tilesLetter)
{

    char c = ' '; int index
    = 0; boolean updated =
    false;

    // cast the source of the click to a LetterTile object
    tilesLetter tilePressed = (tilesLetter) source; c =
    tilePressed.guess();

    // reveal each instance of the character if it appears in
    // the password while ((index =
    password.toLowerCase().indexOf(c, index)) != -1)
    { passwordHidden.setCharAt(index, password.charAt(index));
        index++; updated = true;
    }

    // if the guess was correct, update the GameBoard and check
    //     for a win
    if (updated)
    { correct.setText("Word: " + passwordHidden.toString());

        if (passwordHidden.toString().equals(password))
        { gameRack.removeListeners();
            gameHangman.winningScreen()
            ; newGameDialog();
        }
    }

    // otherwise, add an incorrect guess and check for a loss
    else { incorrect.setText("Incorrect: " + ++numIncorrect);
        if (numIncorrect >= noOfIncorrect)
        { gameHangman.losingScreen();
            gameRack.removeListeners()
            ; newGameDialog();
        }

        else
            gameHangman.nextImage(numIncorrect);
    }
}
```

```
}

    public void mouseClicked(MouseEvent e) {}
    public void mouseReleased(MouseEvent e) {}
    public void mouseEntered(MouseEvent e) {}
    public void mouseExited(MouseEvent e) {}
```