Hangman Project

Table of Contents

# Test-driven Development (TDD)

This project has been built following a Test-Driven Development approach. This approach utilizes planned test cases that must be passed before continuing to the next test case. Throughout the project code will be developed to pass each test case one after another, however it is likely that with the development of code for new test cases, older tests will have to be run again. Therefore, to reduce the amount of manual testing, the implementation of pytest will be used to automatically test the previously passed test cases.

## Development before beginning test cases

I want this application to utilize a UI for display and input, therefore before I begin developing to pass test cases, I decided to construct a blank Tkinter window.

## Test Cases

The following is the series of test cases in order of their initial development. There is a likely chance that code will need to be refactored throughout development which will be documented here.

1. User can input letters 1 at a time to display
2. Generates a random word to the screen
3. Make words invisible and display as underlines to the screen

Future work:

Base word length on difficulty

# Refactoring

## Code Smells

A code smell is a front-end / surface indication that usually corresponds to a deeper problem in the system. This section will clearly detail found code smells and the implementation of their fix.