



CHARLES DARWIN UNIVERSITY SYDNEY CAMPUS

Haymarket, NSW, Australia

**MASTER OF INFORMATION TECHNOLOGY (SOFTWARE
ENGINEERING)**

[PRT582] – SOFTWARE ENGINEERING PROCESS AND TOOLS

Submitted By:

Amsh Shrestha

[S394695]

Submitted To:

Muhammad Rana

Contents

Introduction	2
Process	3
Conclusion	6
Additional Resources	6

Figures

Figure 1: Test-Driven Development	3
Figure 2: Game Logic Implementation	4
Figure 3: GUI Implementation.....	5
Figure 4: Integration	6

Introduction

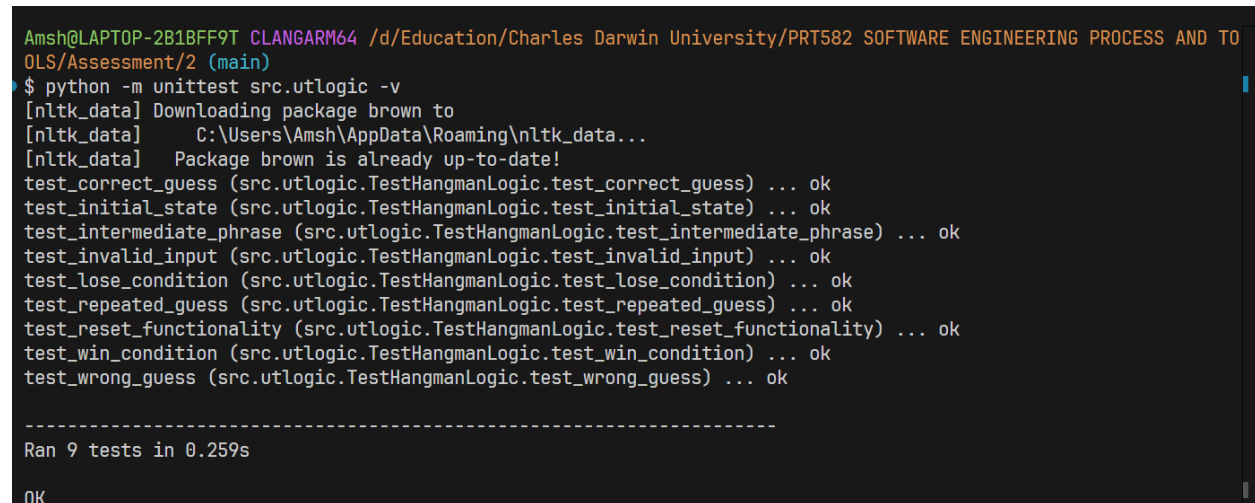
This project implements a Hangman game using Python with a GUI interface. The main objectives are to demonstrate a fully functional game with two levels (basic and intermediate) and to apply Test-Driven Development (TDD) principles to ensure reliability and correctness. Python was chosen due to its simplicity, extensive library support (Tkinter for GUI, NLTK for text processing), and readability, making it ideal for educational and rapid prototyping purposes. The automated unit testing tool used is Python's built-in unittest framework, which allows for structured, repeatable testing of game logic and ensures that new features do not break existing functionality.

Process

1. Test-Driven Development (TDD)

Before implementing game logic, unit tests were written using unittest to cover all major game functionalities, including:

- Correct guess handling
- Wrong guess handling
- Win and lose conditions
- Reset functionality
- Handling invalid inputs and repeated guesses



```
Amsh@LAPTOP-2B1BFF9T CLANGARM64 /d/Education/Charles Darwin University/PRT582 SOFTWARE ENGINEERING PROCESS AND TO  
OLS/Assessment/2 (main)  
$ python -m unittest src.utlogic -v  
[nltk_data] Downloading package brown to  
[nltk_data] C:\Users\Amsh\AppData\Roaming\nltk_data...  
[nltk_data] Package brown is already up-to-date!  
test_correct_guess (src.utlogic.TestHangmanLogic.test_correct_guess) ... ok  
test_initial_state (src.utlogic.TestHangmanLogic.test_initial_state) ... ok  
test_intermediate_phrase (src.utlogic.TestHangmanLogic.test_intermediate_phrase) ... ok  
test_invalid_input (src.utlogic.TestHangmanLogic.test_invalid_input) ... ok  
test_lose_condition (src.utlogic.TestHangmanLogic.test_lose_condition) ... ok  
test_repeated_guess (src.utlogic.TestHangmanLogic.test_repeated_guess) ... ok  
test_reset_functionality (src.utlogic.TestHangmanLogic.test_reset_functionality) ... ok  
test_win_condition (src.utlogic.TestHangmanLogic.test_win_condition) ... ok  
test_wrong_guess (src.utlogic.TestHangmanLogic.test_wrong_guess) ... ok  
  
-----  
Ran 9 tests in 0.259s  
  
OK
```

Figure 1: Test-Driven Development

2. Game Logic Implementation

The core game logic is implemented in `hlogic.py`:

- Basic level: randomly selects words from a dictionary file.
- Intermediate level: generates random phrases from the NLTK Brown corpus.
- Tracks score, remaining tries, guessed letters, and handles game-over conditions.

```
Level selected: basic

--- Starting Basic Level ---
Selected word/phrase: DENMARK
Timer: 15s remaining
Guessed letter: E
Current word: _ E _ _ _ _
Tries left: 6
Score: 1
Game over: False
-----
```

Figure 2: Game Logic Implementation

3. GUI Implementation

The GUI is implemented using Tkinter in `hgui.py`:

- Level selection menu.
- Timer display with auto-decrement.
- Tries and score display.
- Hangman image panel that updates with wrong guesses.
- Alphabet buttons for user input, as well as keyboard input support.
- New game functionality upon game over.

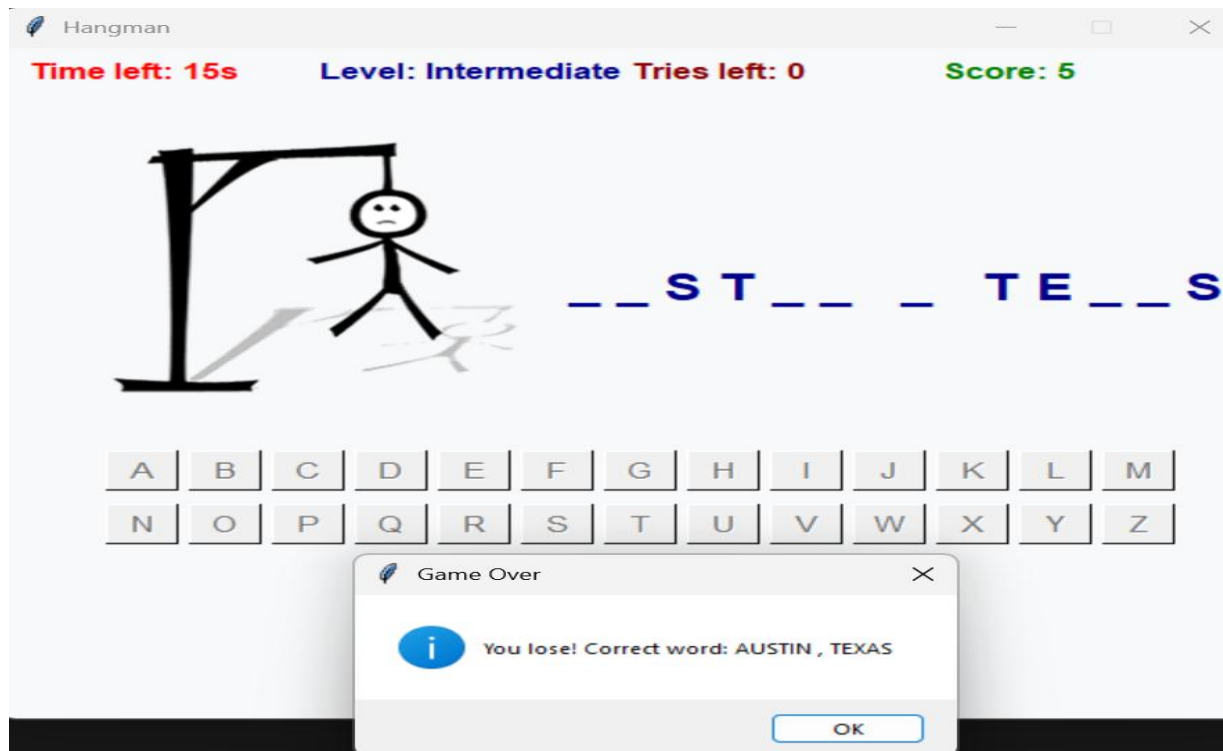


Figure 3: GUI Implementation

4. Integration

- The main program (main.py) initializes the Tkinter root window and starts the Hangman GUI.
- Automated tests ensure all features work correctly before integration.

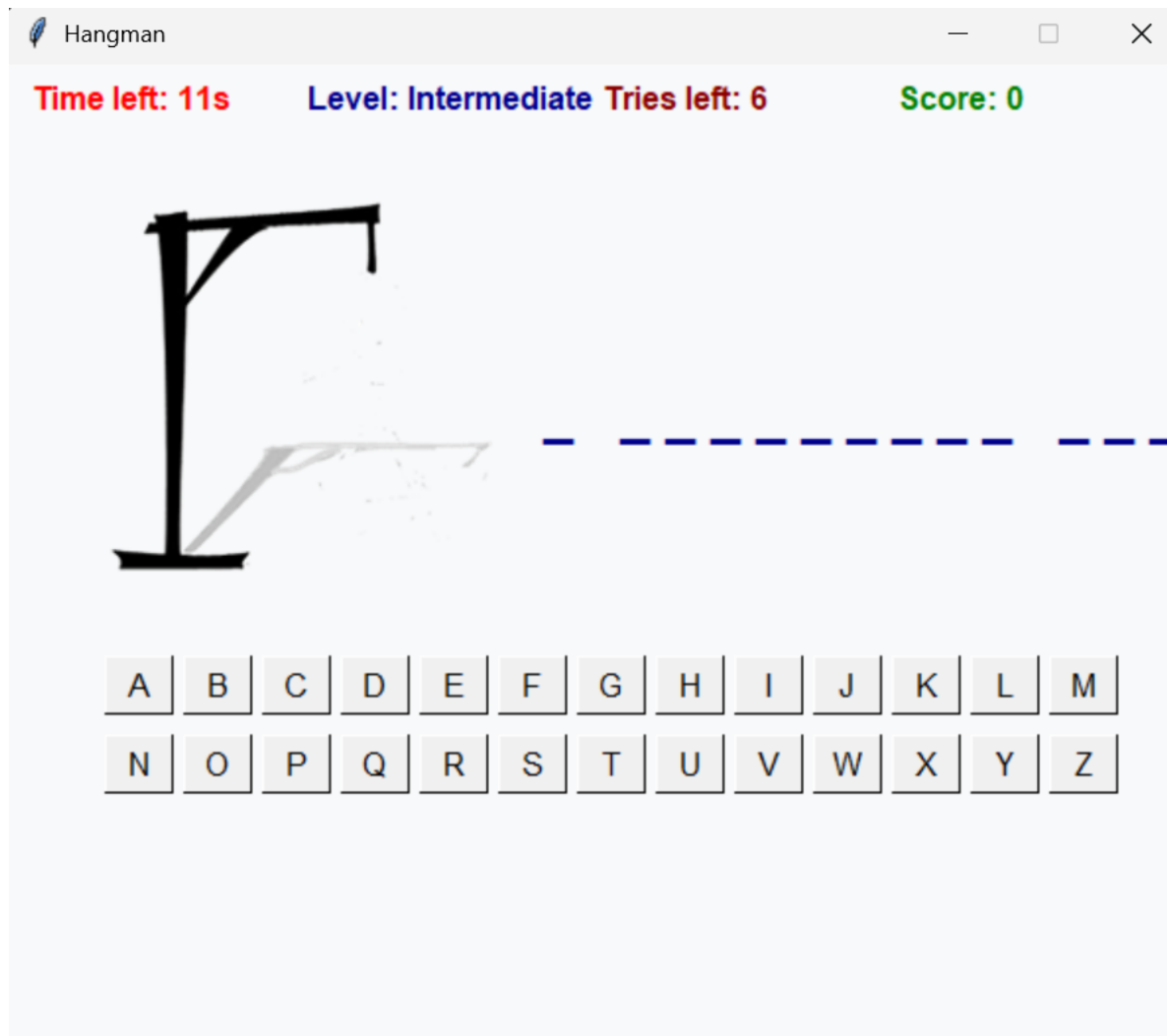


Figure 4: Integration

Conclusion

TDD ensures robustness and reduces bugs during feature development. Separation of logic and GUI improves maintainability and allows for easy testing. Python's Tkinter and NLTK provide simple yet powerful tools for building educational games.

Additional Resources

GitHub Repository

<https://github.com/ams-sth/Hangman-game.git>

Python Files: - hlogic.py (Game logic) - hgui.py (GUI) - utlogic.py (Unit tests) - main.py
(Program entry point)