A white background with blue text

AI-generated content may be incorrect.

**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

[Figures 2](#_Toc207954609)

[Introduction 3](#_Toc207954610)

[Process 3](#_Toc207954611)

[Conclusion 6](#_Toc207954612)

# Figures

[Figure 1: Test-Driven Development 3](#_Toc207954616)

[Figure 2: Game Logic Implementation 4](#_Toc207954617)

[Figure 3: GUI Implementation 4](#_Toc207954618)

[Figure 4: Integration 5](#_Toc207954619)

# Introduction

This project is a Hangman game with two difficulty levels: Basic and Intermediate. In the game, the player has to guess letters to find the hidden word or phrase. The player has 15 seconds to make each guess. If the time runs out, one life is lost. Wrong guesses also cost lives, and correct guesses give points. When the player loses all lives, the correct word is shown. If the player guesses the word correctly, they are congratulated.

I chose Python for this project because it is simple and easy to understand. It also has helpful libraries, like NLTK for text processing and Tkinter for building the GUI. The game logic is tested using Python’s built-in unittest framework. This helps make sure the game works correctly and avoids breaking things when adding new features.

# Process

**Test-Driven Development (TDD)**

I used TDD to plan and build the game logic. First, I wrote unit tests for the main features. These tests included checking correct guesses, wrong guesses, win and lose conditions, reset functionality, invalid input handling, and repeated guesses. Writing tests first helped me catch mistakes early and made coding more reliable.

A computer screen shot of a program

AI-generated content may be incorrect.

Figure 1: Test-Driven Development

**Game Logic Implementation**

The game logic is in hlogic.py. In Basic level, the program randomly selects a word from a dictionary file while in Intermediate level, it generates a random phrase from the NLTK Brown corpus. The logic keeps track of the player’s score, remaining tries, guessed letters, and game-over conditions.

A screen shot of a computer

AI-generated content may be incorrect.

Figure 2: Game Logic Implementation

**GUI Implementation**

The GUI is built with Tkinter in hgui.py. It has a level selection menu, timer display, tries and score labels, hangman images, and buttons for each letter. Players can also use the keyboard to make guesses. After the game ends, a “New Game” button lets players restart and select a new level.

A screenshot of a computer

AI-generated content may be incorrect.

Figure 3: GUI Implementation

**Integration**

The main.py file starts the program by initializing the Tkinter window and launching the GUI. The unit tests ensure that all logic works correctly before connecting it to the interface. Screenshots of tests and gameplay show that all features meet the requirements.

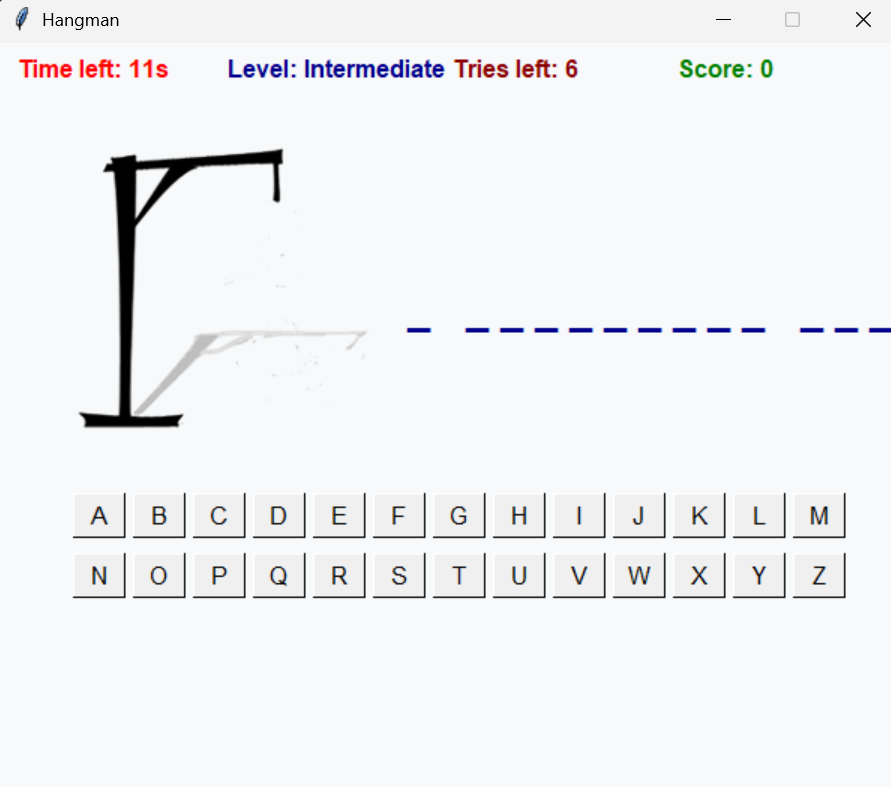


Figure 4: Integration

# Conclusion

This Hangman project helped me understand how to build a program using Test-Driven Development. Writing tests first made it easier to find problems in the game logic. I could handle repeated guesses, invalid inputs, and running out of time without breaking the game.

Separating the logic from the GUI made the code cleaner. I could test the main game without worrying about buttons or the timer. It also makes it easier to add new levels or input methods later.

Working with the timer and keypress input was challenging. I had to make sure the game state and display updated correctly. This helped me learn more about event-driven programming in Tkinter.

Overall, this project taught me good programming habits. Using Git, writing tests, and keeping code organized made development smoother. Python, Tkinter, NLTK, and unittest were enough to build a working and maintainable game.

**GitHub Repository**

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