**Assessment item 1 – Software Unit Testing Report**

**PRT582**

Julio C Del Cid

School of Engineering

Software Engineering Process and Tools

Senior Lecturer Charles Yeo

Due 31 August 2020

Contents

[Assignment 1 - Software Unit Testing Report 3](#_Toc49980905)

[1. The Hangman Game 3](#_Toc49980906)

[The final hangman game 3](#_Toc49980907)

[Functional Requirements (FR) 5](#_Toc49980908)

[FR 1. One word will be generated randomly 5](#_Toc49980909)

[FR 2. Blank spaces representing missing letters 5](#_Toc49980910)

[FR 3. Chosen letter exists in answer 6](#_Toc49980911)

[FR 4. Life % deducted for wrong guess 6](#_Toc49980912)

[FR 5. Player must find the missing words life % becomes zero 6](#_Toc49980913)

[Wrong inputs 7](#_Toc49980914)

[2. Refactoring 8](#_Toc49980915)

[The original hangman game (before refactoring) 8](#_Toc49980916)

[4. Test Driven Development (TDD) 10](#_Toc49980917)

[Description of tests 10](#_Toc49980918)

[5. GIT Repository 11](#_Toc49980919)

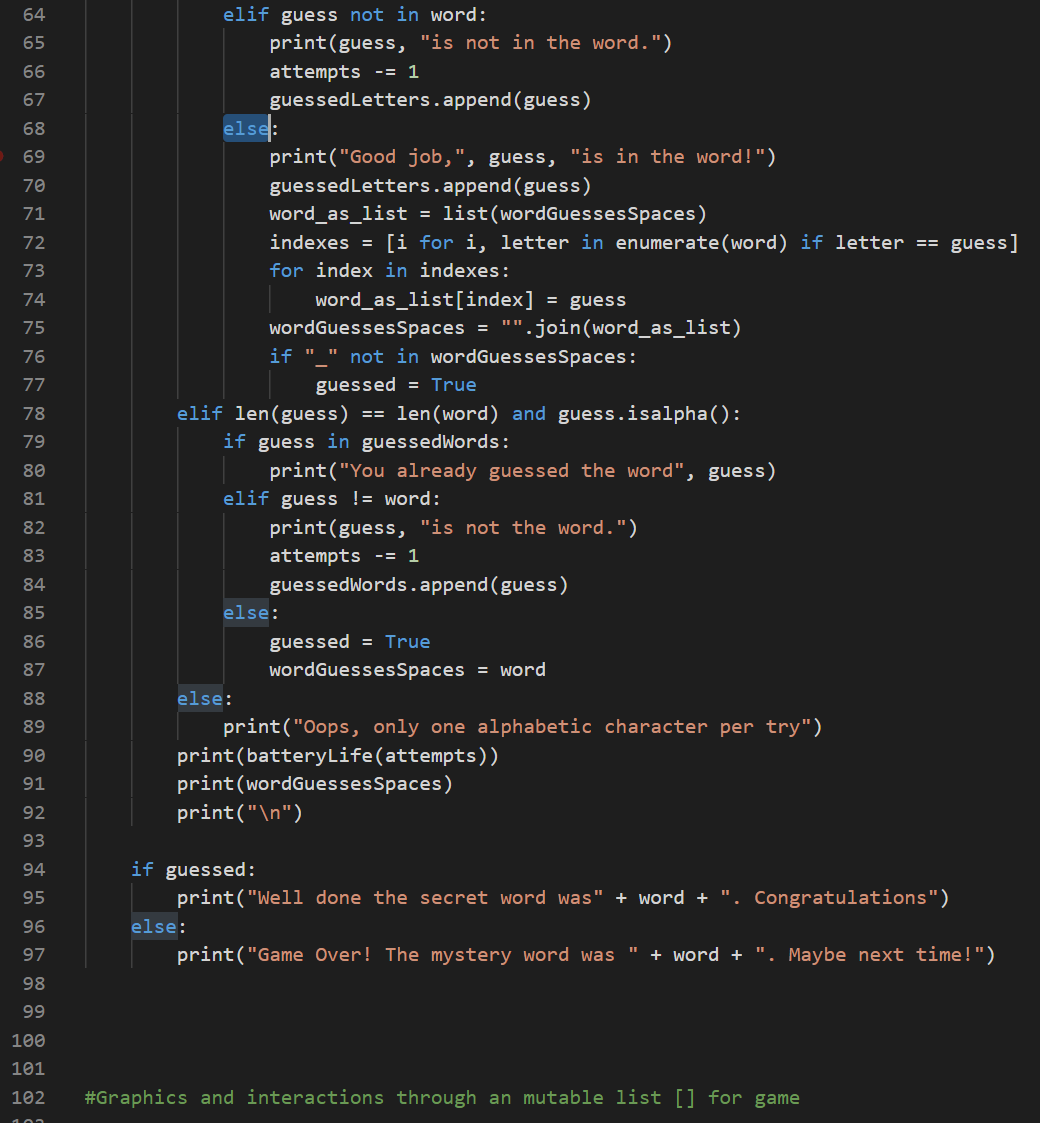
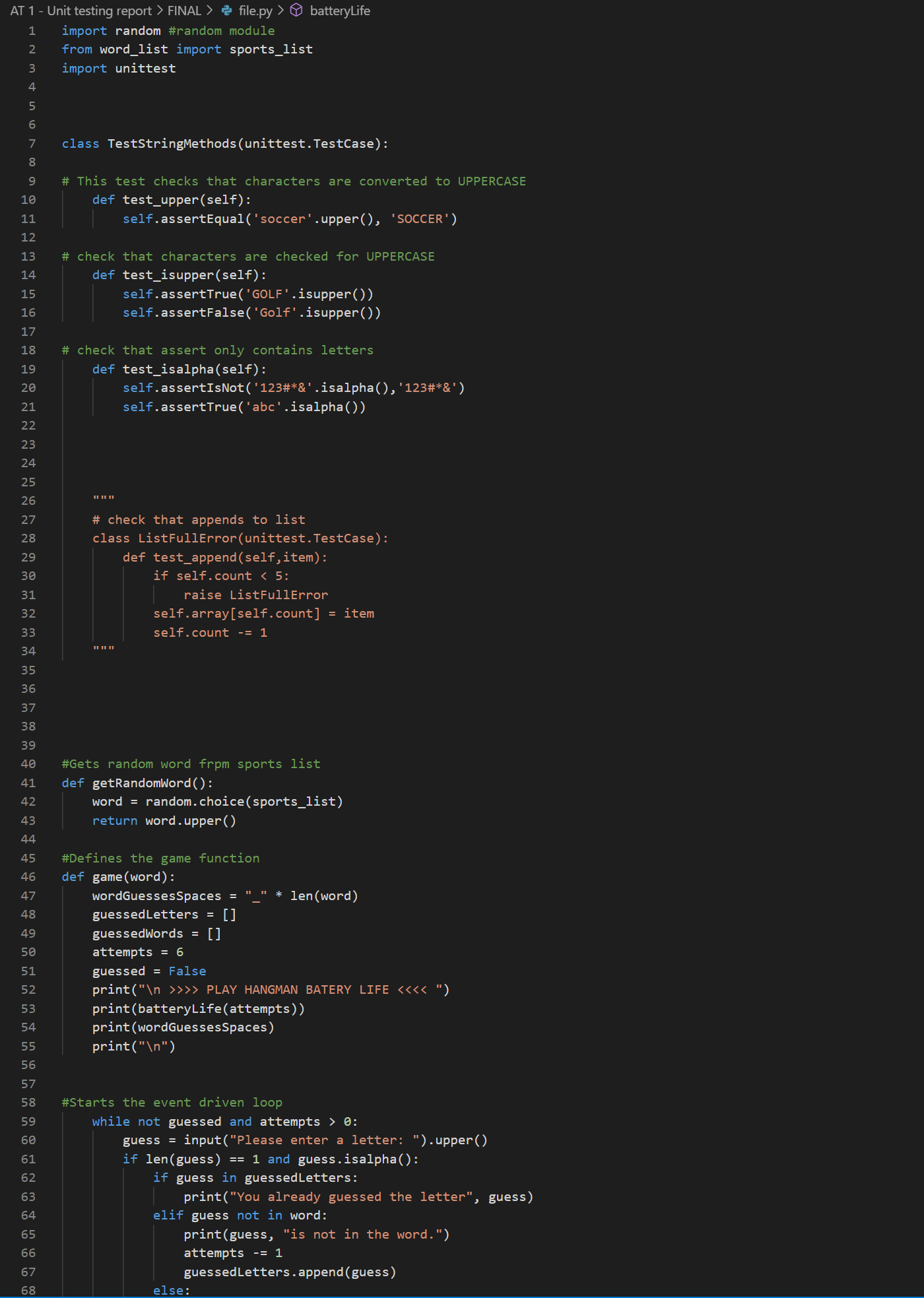
[References 12](#_Toc49980920)

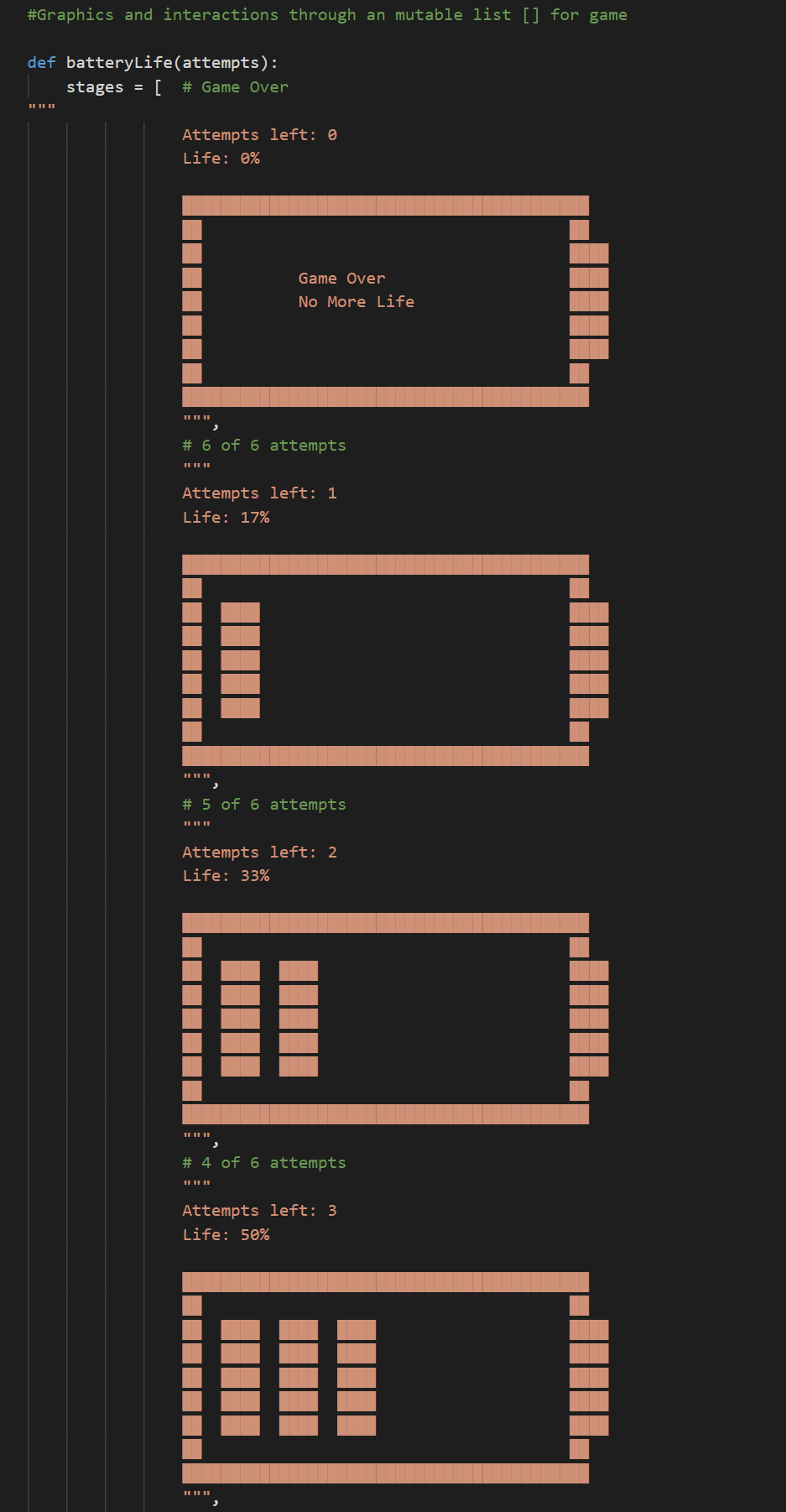
# Assignment 1 - Software Unit Testing Report

# The Hangman Game

## The final hangman game

The final hangman game has programmed and researched through popular game scripts. Methods libraries and other code is popularised in teachings, some of the sources are lists methods and general techniques (New York 2020).





## Functional Requirements (FR)

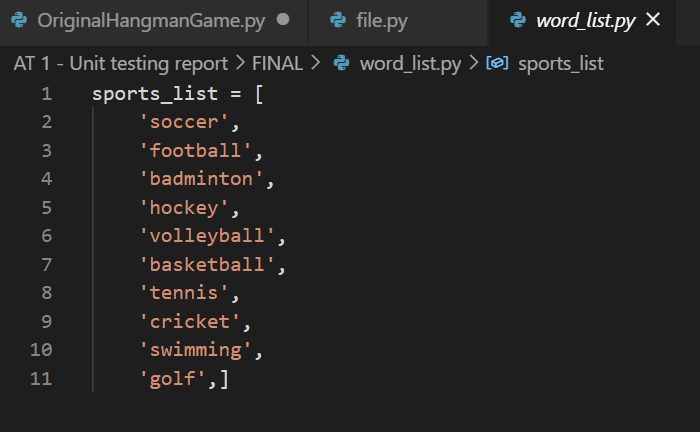
Game functional requirements (FR) are fulfilled and listed below.

## FR 1. One word will be generated randomly

Code for Random function:



The list the random function will cycle through.

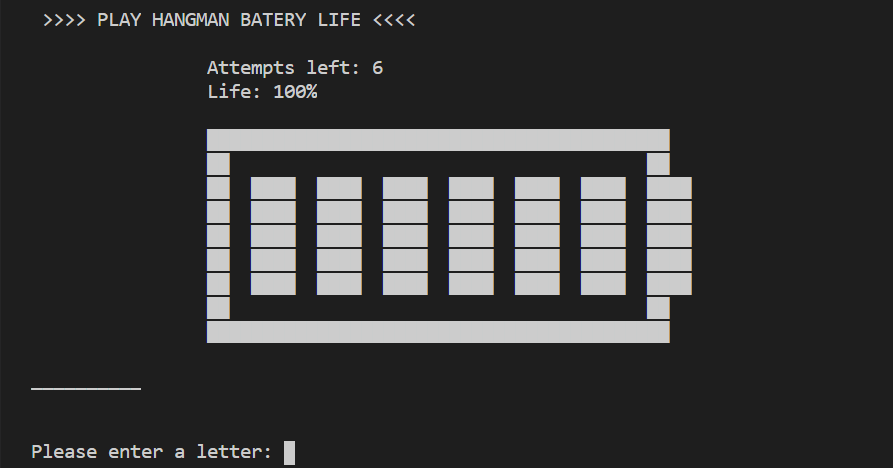


Different attempts offer different words.



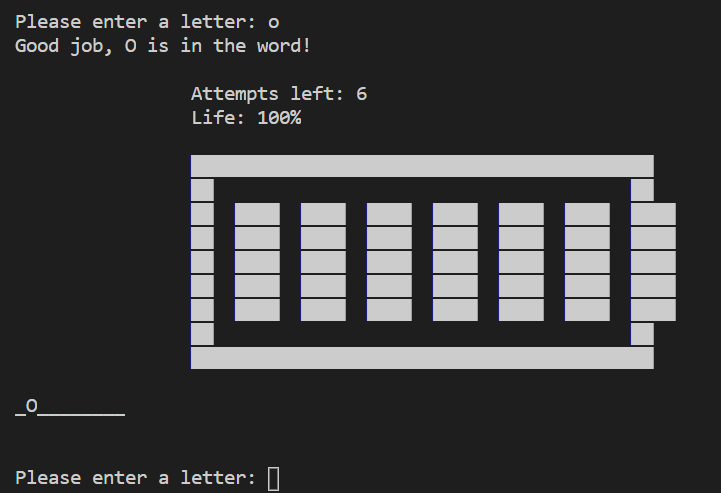
Example of random word generated being SWIMMING.

## FR 2. Blank spaces representing missing letters



Circled area represents the spaces.

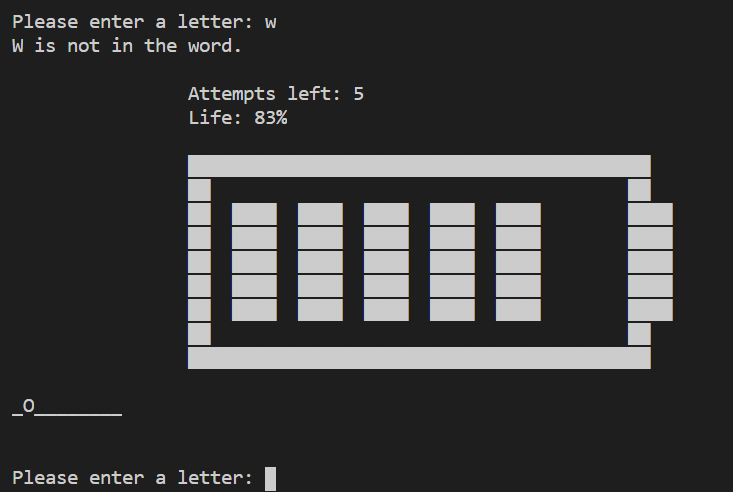
## FR 3. Chosen letter exists in answer



The word above represents “Volleyball”.

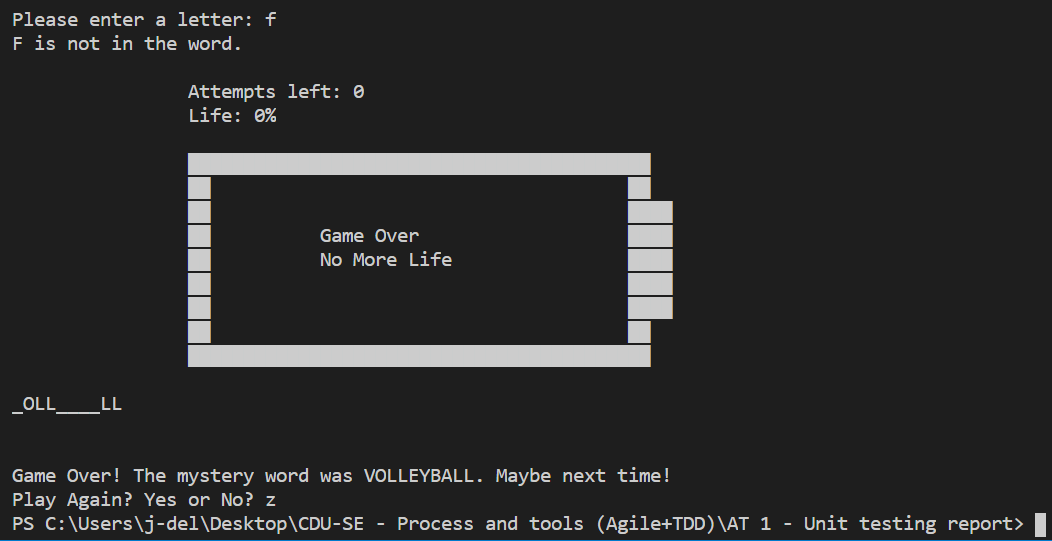
## FR 4. Life % deducted for wrong guess

A % is deducted per wrong answer, which is 100/6 = 16.67% or 17%. The example below demonstrates the error of entering ‘w’.



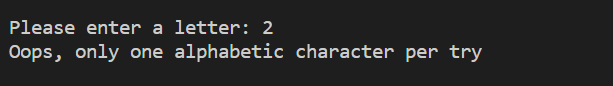
## FR 5. Player must find the missing words life % becomes zero

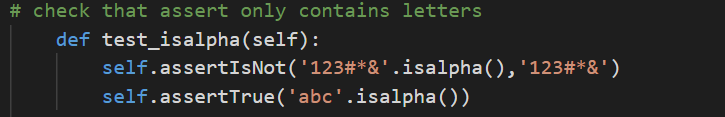
When all attempts have been exhausted life becomes zero represented by “attempts left: 0”.



## Wrong inputs

Wrong inputs are handled by .isalpha string module.





# Refactoring

The original game above is well advanced as the original game lacked many features. The improvements made were:

* Visual display
* Player is presented with blank spaces representing missing letters
* Reduced the attempts to 6 attempts which represents head, body, two arms and two legs
* Checks that characters entered are alphabetic
* Error checks non alphabet characters such as numbers, spaces and special characters
* Makes sure the character entered before is relayed back to the client to try again

Possible future improvements:

* Create a class

Create a API where a word list exists via JSON and the only thing that needs to be updated is the JSON word list.

## The original hangman game (before refactoring)

The hangman game below as a combination of several online tutorials, workshops and resources found online. The main sources used for this game are W3Schools (W3Schools, 2020), GeeksforGeeks (2019) and Stack Exchange (2015).

The script:



The original game had many draw backs and did not meet the requirements, so the game needed to be refactored, which the new code is represented under the title ‘The final hangman game’. Some of the ideas for the script above have been gathered from the following resources: trinket (2020), Invent with Python (2020) and GitHub (2020).

# 4. Test Driven Development (TDD)

Test driven development is something new to me. The tests only test three simple:

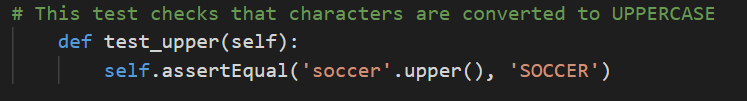
1. test\_upper
2. test\_isupper
3. test\_isalpha

Forcing to think about the game before t is developed was a challenge, however despite this fact the quality of code was apparent immediately. Sources for testing came from Python Software Foundation (2020).

## Description of tests

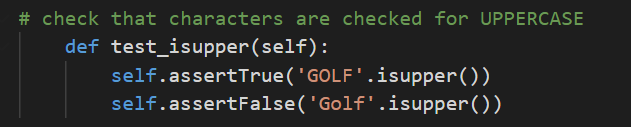
**test\_upper**

Test checks characters are converted to UPPERCASE for easier reading.



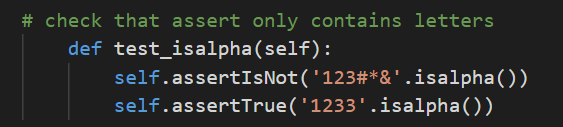
**test\_isupper**

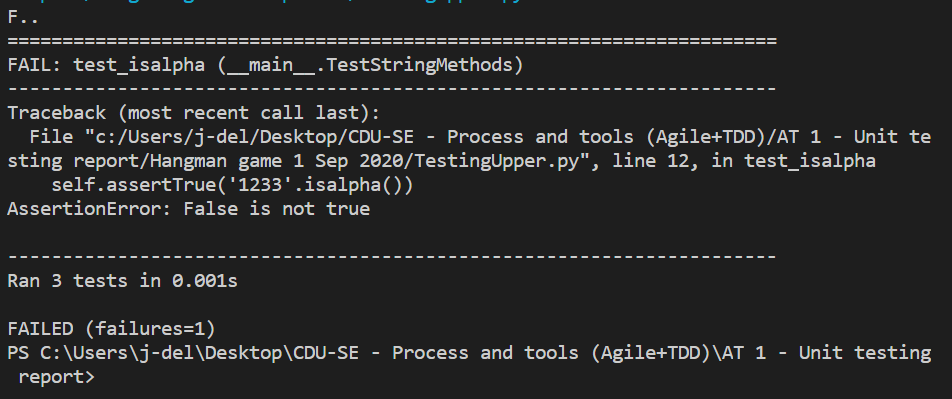
Checks the return values are uppercase.



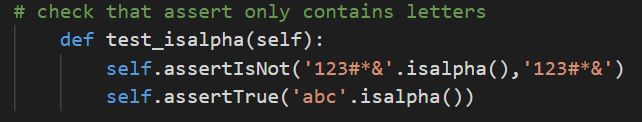
**test\_isalpha**

Tests if the characters entered are from the alphabet.





The tests failed correctly, as the 1234 are not alphabetic characters.



Tests successfully passed.

# 5. GIT Repository

Following is the code repository: <https://github.com/JulioDelCid/prt582-assignment-1>

# References

City of New York University, (2019). Python Lists. <https://academicworks.cuny.edu/cgi/viewcontent.cgi?article=1010&context=si_oers>

Geeks for Geeks, (2019). <https://www.geeksforgeeks.org/>

GitHub. 2020. <https://github.com/kiteco/python-youtube-code/tree/master/build-hangman-in-python>

I Invent with Python. 2020. <https://inventwithpython.com/chapter9.html>

Python Software Foundation. 2020. <https://docs.python.org/3/library/unittest.html#test-cases>

Python Software Foundation. (2020). <https://docs.python.org/3/library/unittest.html#test-cases>

Stack Exchange. (2015). <https://codereview.stackexchange.com/>

Trinket. 2020. <https://trinket.io/python/99f458ee11>

W3Schools. (2020). Python Tutorial. <https://www.w3schools.com/python/default.asp>