******

**School of Mechanical & Manufacturing Engineering (SMME),**

**National University of Science and Technology (NUST),**

Program: BE-Aerospace Section: AE-01

Session: Fall 2023 Semester: 3rd

Course Title: FOP-11

ABDUL MANAN

CMS: 465672

PYTHON PROJECT: **HANGMAN GAME**

# Hangman Code Analysis Report

## 1. Code

import random  
import string  
  
wordlistfile = "words.txt"  
available\_letters = string.ascii\_lowercase   
  
def loadallwords():  
 print("Loading word list from file...")  
 with open(wordlistfile, 'r') as infile:  
 wordslist = infile.read().split()  
 print(len(wordslist), "Words have been loaded.")  
 return wordslist  
  
def availableletters(guessedletters):  
 return ''.join(letter for letter in available\_letters if letter not in guessedletters)  
  
  
def choosewrd(wordlist):  
 return random.choice(wordlist)  
  
def display(tries):  
 stages = [  
 """  
--------  
| |  
|   
|   
|   
|   
-  
""",  
 """  
--------  
| |  
| O  
|   
|   
|   
-  
""",  
 """  
--------  
| |  
| O  
| |  
| |  
|   
-  
""",  
 """  
--------  
| |  
| O  
| \\|  
| |  
|   
-  
""",  
 """  
--------  
| |  
| O  
| \\|/  
| |  
|   
-  
""",  
 """  
--------  
| |  
| O  
| \\|/  
| |  
| /   
-  
""",  
 """  
--------  
| |  
| O  
| \\|/  
| |  
| / \\  
-  
"""  
 ]  
 return stages[tries]  
  
def getguessedwrd(secretwrd, guessedletters):  
 return ''.join(letter if letter in guessedletters else '\_' for letter in secretwrd)  
  
def checkwordguessed(secretwrd, guessedletters):  
 return all(letter in guessedletters for letter in secretwrd)  
  
#This function determins the secret word that the player will have to guess (Using formatting it tells the player how long the word is)  
def hangman(secretwrd):  
 print("Welcome to Hangman!")  
 print(f"The word you have to guess is {len(secretwrd)} letters long.")  
 print("-------------")  
  
 triesleft = 6 #Number of tries left (This will -1 for every wrong try.)  
 guessedletters = []  
  
 while triesleft > 0:  
 print(f"You have {triesleft} tries left.")  
 print("Letters you can use (Available Letters):", availableletters(guessedletters))  
  
 guess = input("Please guess a letter: ").lower()  
 #This line checks if the guessed letter is already been guesses (This will not -1 a try.)  
 if guess in guessedletters:  
 print("Oops! You've already guessed that letter:", getguessedwrd(secretwrd,guessedletters))  
 else:  
 #This Condition cecks if the guessed letter is correct (This will not result in change in tries.)  
 guessedletters.append(guess)  
 if guess in secretwrd:  
 print("Good guess:", getguessedwrd(secretwrd,guessedletters))  
 else:  
 #This condition will -1 a try for the wrongly guessed letter.  
 print("Sorry! That letter is not in the word:", getguessedwrd(secretwrd,guessedletters))  
 triesleft -= 1  
 #This condition will show the current phase of the game.  
 print(display(6-triesleft))  
 print("-------------")  
#This condition will announce if the player has won the game and the game will terminate due to break command  
 if checkwordguessed(secretwrd,guessedletters):  
 print("Congratulations, you won!")  
 break  
  
 if not checkwordguessed(secretwrd, guessedletters):  
 print("Sorry, you ran out of tries. The word was",secretwrd)  
#This function will summon the same number of underscores corresponding to the number of letters  
def numberofunderscores(myword, other\_word):  
 if len(myword) != len(other\_word):  
 return False  
 #This loop will make an array for each element filled with an underscore  
 for i in range(len(myword)):  
 if myword[i] != '\_' and myword[i] != other\_word[i]:  
 return False  
 return True  
  
#This function is used to show every possilble matching word from the word.txt file  
def showpossible(myword, wordlist):  
 matches = [word for word in wordlist if numberofunderscores(myword,word)]  
 if matches:  
 print("Possible word matches:")  
 print(', '.join(matches))  
 else:  
 print("No matches found.")  
  
  
def hangmanwithhints(secretwrd, wordlist):  
 print("Welcome to the game Hangman!")  
 print(f"I am thinking of a word that is {len(secretwrd)} letters long.")  
 print("If you need a hint at any point, enter 'hint'.")  
 print("-------------")  
  
 triesleft = 6  
 guessedletters = []  
  
 while triesleft > 0:  
 print(f"You have {triesleft} tries left.")  
 print("Available letters:", availableletters(guessedletters))  
  
 guess = input("Please guess a letter: ").lower()  
  
 if guess == 'hint':  
 showpossible(getguessedwrd(secretwrd, guessedletters), wordlist)  
 continue  
  
 if guess in guessedletters:  
 print("Oops! You've already guessed that letter:", getguessedwrd(secretwrd, guessedletters))  
 else:  
 guessedletters.append(guess)  
 if guess in secretwrd:  
 print("Good guess:", getguessedwrd(secretwrd, guessedletters))  
 else:  
 if guess in "aeiou":  
 triesleft -= 2  
 else:  
 triesleft -= 1  
 print("Sorry! That letter is not in my word:", getguessedwrd(secretwrd, guessedletters))  
 print(display(6 - triesleft))  
  
 print("-------------")  
  
 if checkwordguessed(secretwrd, guessedletters):  
 print("Congratulations, you won!")  
 break  
  
 if not checkwordguessed(secretwrd, guessedletters):  
 print("Sorry, you ran out of tries. The word was", secretwrd)  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 wordlist = loadallwords()  
 secretword = choosewrd(wordlist)  
  
 option = input("Do you want to play hangman with hints? [yes/no]: ").lower()  
 if option == "yes":  
 hangmanwithhints(secretword, wordlist)  
 elif option == "no":  
 hangman(secretword)

## 2. Code Analysis

The provided Hangman code is a Python implementation of the classic word-guessing game. The code includes the following features:  
  
1. Word Selection: A random word is chosen from the provided word list (`words.txt`).   
2. Gameplay Logic: Players guess letters of a secret word, with feedback provided after each guess.  
3. Hints: Optional gameplay mode includes hints showing possible matches for the current state of the guessed word.  
4. Error Handling: Ensures no penalties for repeated guesses and implements stricter penalties for incorrect vowel guesses.  
5. Visual Feedback: Uses ASCII art to represent the hangman figure at different stages of the game.

## 3. Screenshots

