CS128-5L - PROGRAMMING LANGUAGES FOR DATA SCIENCE LABORATORY
2Q SY2324

NAME: MA. ADDINE ANNE T. CARREON

SECTION: A1

Module Exam 2

Hangman Part 3: The Game with Hints

Instructions:

If you have tried playing Hangman against the computer, you may have noticed that it isn't always easy to beat the computer, especially when it selects an esoteric word (like "esoteric"!). It might be nice if you could ask the computer for a hint, such as a list of all the words that match what you have currently guessed.

For example, if the hidden word is "tact", and you have so far guessed the letter "t", so that you know the solution is "t_ _ t", where you need to guess the two missing letters, it might be nice to know that the set of matching words (at least based on what the computer initially loaded) are:

tact tart taut teat tent test text that tilt tint toot tort tout trot tuft twit

We are going to have you create a variation of Hangman (we call this *hangman_with_hints*, and have provided an initial scaffold for writing it), with the property that if you guess the special character * the computer will find all the words from its loaded list that might match your current guessed word, and print out each of them. Of course, we don't recommend trying this at the first step, since this will print out all 55,900 words that we loaded! But if you are getting close to an answer and are running out of guesses, this might help.

To do this, we are going to ask you to first complete two helper functions:

A. Matching the current guessed word

match_with_gaps takes two parameters: *my_word* and *other_word*. *my_word* is an instance of a guessed word, in other words, it may have some _ 's in places (such as 't_ _ t'). *other_word* is a normal English word.

This function should return *True* if the guessed letters of *my_word* match the corresponding letters of other_word. It should return *False* if the two words are not of the same length or if a guessed letter in *my_word* does not match the corresponding character in *other_word*.

Remember that when a letter is guessed, your code reveals all the positions at which that letter occurs in the secret word. Therefore, the hidden letter (_) cannot be one of the letters in the word that has already been revealed.

Example Usage:

```
match_with_gaps("te_ t", "tact")
```

Instruction

Carreon, Ma. Addine Anne T.
CS128-5L – A1

02/03/2024

```python
import random
import string

WORDLIST_FILENAME = 'words.txt'

def load_words():
    """
    Returns a list of valid words. Words are strings of lowercase letters.

    Depending on the size of the word list, this function may
    take a while to finish.
    """
    print("Loading word list from file...")
    inFile = open(WORDLIST_FILENAME, 'r')
    line = inFile.readline()
    wordlist = line.split()
    print("  ", len(wordlist), "words loaded.")
    return wordlist


def choose_word(wordlist):
    return random.choice(wordlist)


def is_word_guessed(secret_word, letters_guessed):
    return all(letter in letters_guessed for letter in secret_word)


def get_display_word(secret_word, guessed_letters):
    return ''.join(letter if letter in guessed_letters else '_' for letter in secret_word)


def get_guessed_word(secret_word, letters_guessed):
    return ''.join(letter if letter in letters_guessed else '_' for letter in secret_word)


def get_available_letters(letters_guessed):
    return ''.join(letter if letter not in letters_guessed else '' for letter in string.ascii_lowercase)


def match_with_gaps(my_word, other_word):
    return len(my_word) == len(other_word) and all(my_char == other_char or my_char == "_" for my_char,
                                                   other_char in zip(my_word, other_word))


def show_possible_matches(my_word):
    matching_words = [word for word in wordlist if match_with_gaps(my_word, word)]
    if matching_words:
        print("Possible matches:", ', '.join(matching_words))
    else:
        print("No matches found.")
```

```python
def hangman_with_hints(secret_word):
    print("Welcome to Hangman!")
    print("I am thinking of a word that is", len(secret_word), "letters long.")
    guesses_left = 6
    guessed_letters = []

    while guesses_left > 0:
        print("--------------------")
        print("You have", guesses_left, "guesses left.")
        print("Available letters:", get_available_letters(guessed_letters))

        guess = input("Please guess a letter: ").lower()

        if guess == '*':
            show_possible_matches(get_display_word(secret_word, guessed_letters))

        elif guess.isalpha() and len(guess) == 1:
            if guess in guessed_letters:
                print("Oops! You've already guessed that letter. Try again.")
            elif guess in secret_word:
                print("Good guess!")
            else:
                print("Oops! That letter is not in my word.")
                guesses_left -= 1

            guessed_letters.append(guess)
            print(f"Current word: {get_guessed_word(secret_word, guessed_letters)}")

        if is_word_guessed(secret_word, guessed_letters):
            print("Congratulations! You guessed the word:", secret_word)
            break

        if guesses_left == 0:
            print("Sorry, you ran out of guesses. The word was:", secret_word)


if __name__ == "__main__":
    wordlist = load_words()
    secret_word = choose_word(wordlist)
    hangman_with_hints(secret_word)
```

Code

```
Loading word list from file...
   55900 words loaded.

Welcome to Hangman!
I am thinking of a word that is 4 letters long.
--------------------
You have 6 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Oops! That letter is not in my word.
Current word: ____
--------------------
You have 5 guesses left.
Available letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: e
Oops! That letter is not in my word.
Current word: ____
--------------------
You have 4 guesses left.
Available letters: bcdfghijklmnopqrstuvwxyz
Please guess a letter: i
Oops! That letter is not in my word.
Current word: ____
--------------------
You have 3 guesses left.
Available letters: bcdfghjklmnopqrstuvwxyz
Please guess a letter: o
Good guess!
Current word: o___
--------------------
You have 3 guesses left.
Available letters: bcdfghjklmnpqrstuvwxyz
Please guess a letter: u
Good guess!
Current word: o_u_
--------------------
You have 3 guesses left.
Available letters: bcdfghjklmnpqrstvwxyz
Please guess a letter: *
Possible matches: onus, opus, ovum
--------------------
You have 3 guesses left.
Available letters: bcdfghjklmnpqrstvwxyz
Please guess a letter: m
Oops! That letter is not in my word.
Current word: o_u_
--------------------
You have 2 guesses left.
Available letters: bcdfghjklnpqrstvwxyz
Please guess a letter: n
Good guess!
Current word: onu_
--------------------
You have 2 guesses left.
Available letters: bcdfghjklpqrstvwxyz
Please guess a letter: s
Good guess!
Current word: onus
Congratulations! You guessed the word: onus
```

Output