

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

NAME: MA. ADDINE ANNE T. CARREON

SECTION: A1

Laboratory Exercise 5

Hangman Part 1 (Three helper functions)

Instructions:

Before we have you write code to organize the hangman game, we are going to break down the problem into logical subtasks, creating three helper functions you will need to have in order for this game to work. This is a common approach to computational problem solving, and one we want you to begin experiencing.

The file `hangman.py` has a number of already implemented functions you can use while writing up your solution. You can ignore the code in the two functions at the top of the file that have already been implemented for you, though you should understand how to use each helper function by reading the docstrings.

A. Determine whether the word has been guessed

First, implement the function `is_word_guessed` that takes in two parameters a string, `secret_word`, and a list of letters (strings), `letters_guessed`. This function returns a boolean - True if `secret_word` has been guessed (i.e., all the letters of `secret_word` are in `letters_guessed`), and False otherwise. This function will be useful in helping you decide when the hangman game has been successfully completed, and becomes an end-test for any iterative loop that checks letters against the secret word.

For this function, you may assume that all the letters in `secret_word` and `letters_guessed` are lowercase.

Example Usage:

```
secret_word = 'apple'

letters_guessed = ['e', 't', 'k', 'p', 'r', 's']

print(get_guessed_word(secret_word, letters_guessed))

'_ pp_ e'
```

C. Getting all available letters

Next, implement the function `get_available_letters` that takes in one parameter a list of letters, `letters_guessed`. This function returns a string that is comprised of lowercase English letters all lowercase English letters that are not in `letters_guessed`.

This function should return the letters in alphabetical order. For this function, you may assume that all the letters in `letters_guessed` are lowercase.

Hint : You might consider using `string.ascii_lowercase`, which is a string comprised of all lowercase letters:

```
import string

print(string.ascii_lowercase)

abcdefghijklmnopqrstuvwxyz
```

Example Usage:

```
letters_guessed = ['e', 'i', 'k', 'p', 'r', 's']

print get_available_letters(letters_guessed)

abcdefghijklmnopqrstuvwxyz
```

CODE AND OUTPUT:

```
In [1]: def is_word_guessed(secret_word, letters_guessed):
        return all(letter in letters_guessed 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):
            import string
            return ''.join(letter for letter in string.ascii_lowercase if letter not in letters_guessed)

        if __name__ == "__main__":
            secret_word = 'apple'
            letters_guessed = ['e', 'i', 'k', 'p', 'r', 's']

            print(">>>secret_word = 'apple'")
            print(">>>letters_guessed = ['e', 'i', 'k', 'p', 'r', 's']")
            print(">>>print(is_word_guessed(secret_word, letters_guessed))")
            print(is_word_guessed(secret_word, letters_guessed), end='\n\n')

            print(">>>secret_word = 'apple'")
            print(">>>letters_guessed = ['e', 'i', 'k', 'p', 'r', 's']")
            print(">>>print(get_guessed_word(secret_word, letters_guessed))")
            print('\'' + get_guessed_word(secret_word, letters_guessed) + '\'', end='\n\n')

            print(">>>letters_guessed = ['e', 'i', 'k', 'p', 'r', 's']")
            print(">>>print(get_available_letters(letters_guessed))")
            print(get_available_letters(letters_guessed))

>>>secret_word = 'apple'
>>>letters_guessed = ['e', 'i', 'k', 'p', 'r', 's']
>>>print(is_word_guessed(secret_word, letters_guessed))
False

>>>secret_word = 'apple'
>>>letters_guessed = ['e', 'i', 'k', 'p', 'r', 's']
>>>print(get_guessed_word(secret_word, letters_guessed))
'_ pp_e'

>>>letters_guessed = ['e', 'i', 'k', 'p', 'r', 's']
>>>print(get_available_letters(letters_guessed))
abcdefghijklmnopqrstuvwxyz
```