HANGMAN PART 1: IS THE WORD GUESSED? : 5.0 POINTS

Please read the Hangman Introduction before starting this problem. The helper functions you will be creating in the next three exercises are simply suggestions. If you'd prefer to structure your Hangman program in a different way, feel free to skip these exercises. However, if you're new to programming, or at a loss of how to construct this program, we strongly suggest that you implement the helper functions on this page before continuing on to Hangman Part 2.

We'll start by writing 3 simple functions that will help us easily code the Hangman problem. First, implement the function `isWordGuessed` that takes in two parameters - a string, `secretWord`, and a list of letters, `lettersGuessed`. This function returns a boolean - True if `secretWord` has been guessed (ie, all the letters of `secretWord` are in `lettersGuessed`) and False otherwise.

Example Usage:

```
>>> secretWord = 'apple'
>>> lettersGuessed = ['e', 'i', 'k', 'p', 'r', 's']
>>> print isWordGuessed(secretWord, lettersGuessed)
False
```

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

```
def isWordGuessed(secretWord, lettersGuessed):
    '''
    secretWord: string, the word the user is guessing
    lettersGuessed: list, what letters have been guessed so far
    returns: boolean, True if all the letters of secretWord are in lettersGuessed;
      False otherwise
    '''
    # FILL IN YOUR CODE HERE...

```

Unsubmitted

Check Save *You have used 0 of 30 submissions*

PRINTING OUT THE USER'S GUESS : 5.0 POINTS

Next, implement the function `getGuessedWord` that takes in two parameters - a string, `secretWord`, and a list of letters, `lettersGuessed`. This function returns a string that is comprised of letters and underscores, based on what letters in `lettersGuessed` are in `secretWord`. This shouldn't be too different from `isWordGuessed`!

Example Usage:

```
>>> secretWord = 'apple'
>>> lettersGuessed = ['e', 'i', 'k', 'p', 'r', 's']
>>> print getGuessedWord(secretWord, lettersGuessed)
'_ pp_ e'
```

When inserting underscores into your string, it's a good idea to add at least a space after each one, so it's clear to the user how many unguessed letters are left in the string (compare the readability of `____` with `_ _ _ _`). However, you are free to use spacing in any way you wish - our grader will only check that the letters and underscores are in the proper order; it will not look at spacing.

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

```
1 def getGuessedWord(secretWord, lettersGuessed):
2     '''
3     secretWord: string, the word the user is guessing
4     lettersGuessed: list, what letters have been guessed so far
5     returns: string, comprised of letters and underscores that represents
6       what letters in secretWord have been guessed so far.
7     '''
8     # FILL IN YOUR CODE HERE...
9
```

Unsubmitted

Check **Save** *You have used 0 of 30 submissions*

PRINTING OUT ALL AVAILABLE LETTERS : 5.0 POINTS

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

Example Usage:

```
>>> lettersGuessed = ['e', 'i', 'k', 'p', 'r', 's']
>>> print getAvailableLetters(lettersGuessed)
abcdfghjlmnoqtuvwxyz
```

For this function, you may assume that all the letters in `lettersGuessed` 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
```

```
def getAvailableLetters(lettersGuessed):
    '''
    lettersGuessed: list, what letters have been guessed so far
    returns: string, comprised of letters that represents what letters have not
      yet been guessed.
    '''
    # FILL IN YOUR CODE HERE...

```

Unsubmitted

Check **Save** *You have used 0 of 30 submissions*

Show Discussion **New Post**