Hangman Part 2 - The Game / Complex Hangman | Problem Set 3 | Contenu du cours 6.00.1x

courses.edx.org/courses/course-v1:MITx+6.00.1x_6+2T2015/courseware/sp13_Week_3/sp13_Problem_Set_3/

Now you will implement the function `hangman`, which takes one parameter - the `secretWord` the user is to guess. This starts up an interactive game of Hangman between the user and the computer. Be sure you take advantage of the three helper functions, `isWordGuessed`, `getGuessedWord`, and `getAvailableLetters`, that you've defined in the previous part.

Hints:

- You should start by noticing where we're using the provided functions (at the top of `ps3_hangman.py`) to load the words and pick a random one. Note that the functions `loadWords` and `chooseWord` should only be used on your local machine, not in the tutor. When you enter in your solution in the tutor, you only need to give your `hangman` function.
- Consider using `lower()` to convert user input to lower case. For example:

  ```
  guess = 'A'
  guessInLowerCase = guess.lower()
  ```
- Consider writing additional helper functions if you need them!
- There are four important pieces of information you may wish to store:

  1. `secretWord`: The word to guess.
  2. `lettersGuessed`: The letters that have been guessed so far.
  3. `mistakesMade`: The number of incorrect guesses made so far.
  4. `availableLetters`: The letters that may still be guessed. Every time a player guesses a letter, the guessed letter must be removed from `availableLetters` (and if they guess a letter that is not in `availableLetters`, you should print a message telling them they've already guessed that - so try again!).

Sample Output

The output of a winning game should look like this...

```
Loading word list from file...
55900 words loaded.
Welcome to the game, Hangman!
I am thinking of a word that is 4 letters long.
-------------
You have 8 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Good guess: _ a_ _
------------
You have 8 guesses left.
```

```
Available letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Oops! You've already guessed that letter: _ a_ _
------------
You have 8 guesses left.
Available letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: s
Oops! That letter is not in my word: _ a_ _
------------
You have 7 guesses left.
Available letters: bcdefghijklmnopqrtuvwxyz
Please guess a letter: t
Good guess: ta_ t
------------
You have 7 guesses left.
Available letters: bcdefghijklmnopqruvwxyz
Please guess a letter: r
Oops! That letter is not in my word: ta_ t
------------
You have 6 guesses left.
Available letters: bcdefghijklmnopquvwxyz
Please guess a letter: m
Oops! That letter is not in my word: ta_ t
------------
You have 5 guesses left.
Available letters: bcdefghijklnopquvwxyz
Please guess a letter: c
Good guess: tact
------------
Congratulations, you won!
```

And the output of a losing game should look like this...

```
Loading word list from file...
55900 words loaded.
Welcome to the game Hangman!
I am thinking of a word that is 4 letters long.
-----------
You have 8 guesses left.
Available Letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Oops! That letter is not in my word: _ _ _ _
-----------
You have 7 guesses left.
Available Letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: b
Oops! That letter is not in my word: _ _ _ _
-----------
You have 6 guesses left.
Available Letters: cdefghijklmnopqrstuvwxyz
```

```
Please guess a letter: c
Oops! That letter is not in my word: _ _ _ _
-----------
You have 5 guesses left.
Available Letters: defghijklmnopqrstuvwxyz
Please guess a letter: d
Oops! That letter is not in my word: _ _ _ _
-----------
You have 4 guesses left.
Available Letters: efghijklmnopqrstuvwxyz
Please guess a letter: e
Good guess: e_ _ e
-----------
You have 4 guesses left.
Available Letters: fghijklmnopqrstuvwxyz
Please guess a letter: f
Oops! That letter is not in my word: e_ _ e
-----------
You have 3 guesses left.
Available Letters: ghijklmnopqrstuvwxyz
Please guess a letter: g
Oops! That letter is not in my word: e_ _ e
-----------
You have 2 guesses left.
Available Letters: hijklmnopqrstuvwxyz
Please guess a letter: h
Oops! That letter is not in my word: e_ _ e
-----------
You have 1 guesses left.
Available Letters: ijklmnopqrstuvwxyz
Please guess a letter: i
Oops! That letter is not in my word: e_ _ e
-----------
Sorry, you ran out of guesses. The word was else.
```

Note that if you choose to use the helper functions `isWordGuessed`, `getGuessedWord`, or `getAvailableLetters`, you do not need to paste your definitions in the box. We have supplied our implementations of these functions for your use in this part of the problem. If you use additional helper functions, you will need to paste those definitions here.

Your function should include calls to `raw_input` to get the user's guess.

Why does my Output Have `None` at Various Places?

`None` is a keyword and it comes from the fact that you are printing the result of a function that does not return anything. For example:

```
    def foo(x):
        print x
```

If you just call the function with `foo(3)`, you will see output:

```
3   #-- because the function printed the variable
```

However, if you do `print foo(3)`, you will see output:

```
3     #-- because the function printed the variable
None  #-- because you printed the function (and hence the return)
```

All functions return something. If a function you write does not return anything (and just prints something to the console), then the default action in Python is to `return None`

Function call: hangman(c)

```
Testing if we can correctly guess a short word...
```

Output:

```
Welcome to the game, Hangman!
I am thinking of a word that is 1 letters long.
-------------
You have 8 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: c
Good guess: c
-------------
Congratulations, you won!
None
```

Function call: hangman(zzz)

```
Testing if we can correctly fill in repeat letters ...
```

Output:

```
Welcome to the game, Hangman!
I am thinking of a word that is 3 letters long.
-------------
You have 8 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: z
Good guess: zzz
-------------
Congratulations, you won!
None
```

Function call: hangman(c)

```
Testing if we can incorrectly guess a short word...
```

Output:

```
Welcome to the game, Hangman!
I am thinking of a word that is 1 letters long.
-------------
You have 8 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Oops! That letter is not in my word: _
-------------
You have 7 guesses left.
Available letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: b
Oops! That letter is not in my word: _
-------------
You have 6 guesses left.
Available letters: cdefghijklmnopqrstuvwxyz
Please guess a letter: d
Oops! That letter is not in my word: _
-------------
You have 5 guesses left.
Available letters: cefghijklmnopqrstuvwxyz
Please guess a letter: e
Oops! That letter is not in my word: _
-------------
You have 4 guesses left.
Available letters: cfghijklmnopqrstuvwxyz
Please guess a letter: f
Oops! That letter is not in my word: _
-------------
You have 3 guesses left.
Available letters: cghijklmnopqrstuvwxyz
Please guess a letter: g
Oops! That letter is not in my word: _
-------------
You have 2 guesses left.
Available letters: chijklmnopqrstuvwxyz
Please guess a letter: h
Oops! That letter is not in my word: _
-------------
You have 1 guesses left.
Available letters: cijklmnopqrstuvwxyz
Please guess a letter: i
Oops! That letter is not in my word: _
-------------
Sorry, you ran out of guesses. The word was else.
None
```

Function call: hangman(sea)

```
Testing if we handle repeat correct guesses...
```

Output:

```
Welcome to the game, Hangman!
I am thinking of a word that is 3 letters long.
-------------
You have 8 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: a
Good guess: _ _ a
-------------
You have 8 guesses left.
Available letters: bcdefghijklmnopqrstuvwxyz
Please guess a letter: e
Good guess: _ ea
-------------
You have 8 guesses left.
Available letters: bcdfghijklmnopqrstuvwxyz
Please guess a letter: a
Oops! You've already guessed that letter: _ ea
-------------
You have 8 guesses left.
Available letters: bcdfghijklmnopqrstuvwxyz
Please guess a letter: e
Oops! You've already guessed that letter: _ ea
-------------
You have 8 guesses left.
Available letters: bcdfghijklmnopqrstuvwxyz
Please guess a letter: s
Good guess: sea
-------------
Congratulations, you won!
None
```

Function call: hangman(y)

```
Testing if we handle repeat incorrect guesses...
```

Output:

```
Welcome to the game, Hangman!
I am thinking of a word that is 1 letters long.
-------------
You have 8 guesses left.
Available letters: abcdefghijklmnopqrstuvwxyz
Please guess a letter: x
Oops! That letter is not in my word: _
-------------
You have 7 guesses left.
Available letters: abcdefghijklmnopqrstuvwyz
```

```
    Please guess a letter: z
    Oops! That letter is not in my word: _
    -------------
    You have 6 guesses left.
    Available letters: abcdefghijklmnopqrstuvwy
    Please guess a letter: x
    Oops! You've already guessed that letter: _
    -------------
    You have 6 guesses left.
    Available letters: abcdefghijklmnopqrstuvwy
    Please guess a letter: z
    Oops! You've already guessed that letter: _
    -------------
    You have 6 guesses left.
    Available letters: abcdefghijklmnopqrstuvwy
    Please guess a letter: y
    Good guess: y
    -------------
    Congratulations, you won!
    None
```

To make the output less verbose, we are only testing simple input cases here. When your hangman function passes these checks, paste the same code again into the next box to test it on harder input cases.