

## HANGMAN PART 2: THE GAME (15 满分)

感谢 GLHEZJNUCN 童鞋的给力翻译!!

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.

现在你将开始实现（完整的游戏）程序hangman, 它有一个输入参数 - 用户需要去猜测的单词 `secretWord`（后续程序用到的变量名）。由它开始游戏程序Hangman用户与计算机的交互过程。确保你用到之前部分设计完成的辅助函数`isWordGuessed`, `getGuessedWord`, 和 `getAvailableLetters`。

### Hints 提示:

- 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.

你需要注意到我们何时用到提供的函数 (程序 `ps3_hangman.py`的开始处) 来装入词库以及随机取词. 注意函数`loadWords` 和 `chooseWord` 只能用于你本地机器测试时, 在提交代码中不用到. 当你提交解答代码时, 你只须给出函数`hangman`.

- Consider using `lower()` to convert user input to lower case. For example:考虑用函数 `lower()` 来将用户的输入转换为小写。

```
guess = 'A'
guessInLowerCase = guess.lower()
```

- Consider writing additional helper functions if you need them!如果你需要, 你可以考虑编写额外的辅助函数
- There are four important pieces of information you may wish to store:4个重要的信息你可能需要考虑保存
  - `secretWord`: The word to guess.猜测的目标单词.
  - `lettersGuessed`: The letters that have been guessed so far.追踪用户猜测过的字母
  - `mistakesMade`: The number of incorrect guesses made so far.记录用户猜测错误的次数
  - `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!).还可以用来被猜测的字母, 用户猜测过的字母需要从`availableLetters`移除, 用户重复猜了的字母, 你需要告知用户你已经猜过这个字母。

### Sample Output 输出样例

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

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

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.

注意你如果用到了辅助函数`isWordGuessed`, `getGuessedWord`, 或者`getAvailableLetters`, 你无需将他们的代码复制到这儿的代码窗口。这一部分我们会提供这些函数让你直接使用。但如果你用了其他的辅助函数, 你需要将代码也复制在这儿的代码窗口。

Your function should include calls to `raw_input` to get the user's guess.你的函数需要用 `raw_input` 来获得用户的每次猜测。

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`

```

7  * At the start of the game, let the user know how many
8  letters the secretWord contains.
9
10 * Ask the user to supply one guess (i.e. letter) per round.
11
12 * The user should receive feedback immediately after each guess
13 about whether their guess appears in the computers word.
14
15 * After each round, you should also display to the user the
16 partially guessed word so far, as well as letters that the
17 user has not yet guessed.
18
19 Follows the other limitations detailed in the problem write-up.
20 """
21 # FILL IN YOUR CODE HERE...
```

未答复

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.为使我们的检测输出不至于冗长, 这里我们仅用简单的情形做测试, 当你的代码通过了这个测试, 将代码复制到下面的窗口, 进行更为苛刻的测试。

提交

保存

## HANGMAN - COMPLEX TESTS (20 满分)


```
1 # When your hangman function passes the checks in the previous
2 # box, paste your function definition here to test it on harder
3 # input cases.
4
5
```

未答复

提交

保存

显示讨论

 新的帖子

Copyright 2013-2017 北京慕华信息科技有限公司

京ICP证140571号 | 京公网安备 11010802017721 广播电视节目制作经营许可证（京）字第05791号

POW  
OI