**Computer Science Principles and Programming**

**Project - Hangman**

For this problem, you will implement a variation of the classic word game Hangman. For those of you who are unfamiliar with the rules, you may read all about it [here](https://en.wikipedia.org/wiki/Hangman_(game)). In this problem, the second player will always be the computer, who will be picking a word at random.

In this problem, you will implement a function, called hangman, that will start up and carry out an interactive Hangman game between a player and the computer. Before we get to this function, we'll first implement a few helper functions to get you going.

For this problem, you will need the file words.txt. This file contains the list of words that can be used in this game.

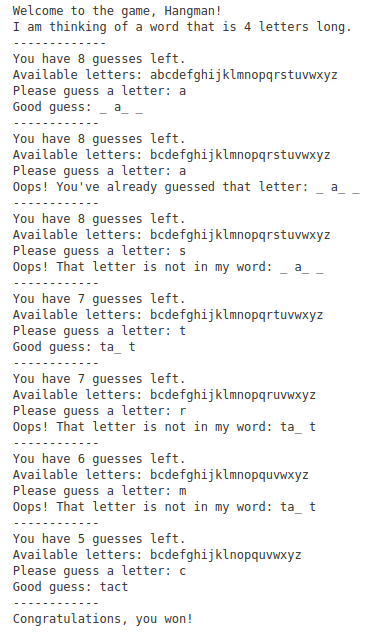
**Requirements**

Here are the requirements for your game:

1. The computer must select a word at random from the list of available words that was provided in words.txt. The functions for loading the word list and selecting a random word have to be created.
2. The game must be interactive; the flow of the game should go as follows:
   1. At the start of the game, let the user know how many letters the computer's word contains.
   2. Ask the user to supply one guess (i.e. letter) per round.
   3. The user should receive feedback immediately after each guess about whether their guess appears in the computer's word.
   4. After each round, you should also display to the user the partially guessed word so far, as well as letters that the user has not yet guessed.
3. Some additional rules of the game:
4. A user is allowed 6 guesses. Make sure to remind the user of how many guesses s/he has left after each round. Assume that players will only ever submit one character at a time (A-Z).
5. A user loses a guess only when s/he guesses incorrectly.
6. If the user guesses the same letter twice, do not take away a guess - instead, print a message letting them know they've already guessed that letter and ask them to try again.
7. The game should end when the user constructs the full word or runs out of guesses. If the player runs out of guesses (s/he "loses"), reveal the word to the user when the game ends.

**Sample Output**

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



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

