Problem Description

This problem was borrowed from Google's Python class. You are going to write a program that can count words that show up in the novel *Alice in Wonderland*. You will begin by downloading a text file that contains the entirety of the novel and a template Python script. To make access to each of these easy, I'm currently hosting both of these files on one of Prof. Murphey's lab servers. You can directly download them with the following two commands:

- `wget http://robotics.mech.northwestern.edu/~jarvis/wordcount.py`
- `wget http://robotics.mech.northwestern.edu/~jarvis/alice.txt`

Goals

1. The template comes with a `main` function that already parses some command line input and then calls the appropriate functions. Your goal is to implement these functions to provide the following behaviors:
   - If someone passes the `--count` flag i.e. `python wordcount.py --count alice.txt` the `print_words` function should count how often each word appears in the text and print out the words and their counts. Further details are found in the header of `wordcount.py`.
   - If someone passes the `--topcount` flag i.e. `python wordcount.py --topcount alice.txt`, this should print out a sorted list of the 20 most common words. More details are in the `wordcount.py` file.

Problem Extensions

1. The simple implementation of goal 1 uses very easy string matching to detect unique words. So in my initial implementation the words `yourself`, `yourself!'`, `yourself,`, `yourself,'`, and `yourself.'` all appeared as separate words. One could imagine adding many rules to deal with these kinds of cases, but it would be quite tedious. The answer is to use the *Regular Expressions* Python module (just called `re` in Python). This module can automatically strip punctuation characters. Try playing with this module and see if you can make your code from above more robust.

Useful Notes

1. You will likely want to use a dictionary where the keys are the words in the text, and the items are the count.
2. You can use the `open` command to open a file, but it is good practice to always close the file before your script finishes.

3. Python uses very convenient syntax to read in files. If `file` is already an open file then the command `for lines in file:` will iterate through the whole file and each time, the variable "lines" will be a line in the file. Then simple commands can split the line into words. For example, read the documentation on the built-in string function `split`.

4. The command `sorted` can produce sorted copies of data.