Homework #2 **Due: 10/4**

This homework will test your ability to make use of Python's built-in data structures. A bad design will waste your time coding and lead to buggy and hard-to-test code.

1. [50 points] Write a function `concordance(f, unique=True)` which reads a stream (opened file) `f` and produces a *concordance*: a dictionary keyed by every word in the stream. The value associated with each key is a list whose elements are the numbers of the lines within the stream on which the name appears. There is one entry in the sequence for each instance of the word. The output of `concordance()` does not involve the name of the input stream, if it even has one (e.g. `f` could, for instance, be `sys.stdin`).

   If the `unique` parameter is true, the entries are unique: If the word appears more than once on the same line of the same file, there is still only one entry. If the flag is false, redundant entries are counted. (The latter might be used to compute word frequencies or as in the second part of this assignment.)

   Put `concordance()` in its own module `concordance.py` and include a thorough "`if __name__ == '__main__':`" self-test.

   Notes:

   - Words are treated caselessly: All input text should be converted to lower case.
   - Your code should ignore punctuation, digits, and other non-letters, except that hyphens and apostrophes should be accepted as part of a word if they occur "inside" a word, so "cock-o'-the-walk" is one word. (Don't special case this word. There are others in the test suite.)
   - You may assume your input is ASCII characters only. Don't worry about what to do with arbitrary Unicode.

2. [50 points] Write a Python script (*not* a module) `concord` that compiles concordances for every file specified on its command line, merges them into one big one, and then prints them out in alphabetical order of the keys. Each key should be followed by the total number of instances, then one line for each file name with the line numbers within each file. Lines that contain more than one instance should be followed by a count of the number of instances in parentheses.

A typical output should look something like this:

```
coconuts (4):
      holy_grail.txt: 2, 3, 87, 123

maudling (4):
      episode19.txt: 23
      episode20.txt: 45(2), 74

romanes (3):
      brian.txt: 65, 66, 67
```

Notes:

- `concord` must `import` from and use the `concordance` module you turn in for the first part unchanged. `concord` will need to adapt its output to do what you want to do.
- The words must appear be in alphabetical order.
- Within each word listing, the file names must be in alphabetical order.
- Within each file name for each word listing, the line numbers must be increasing.

[+10 points extra credit] Add a "`-h`" command line option that will output (to `sys.stdout`) an HTML file suitable as input for a web browser, for example:

```
$ concord -h foo.txt >concord_out.html
$ firefox concord_out.html
```

that contains an HTML table (i.e., "`<table> ... </table>`") with the same information. (This would allow `concord` to be incorporated into a web server, but you are not expected to do this.)

Note: No fair using "`<pre> ... </pre>`" tags!