CS2044 Homework 2

Due: Sunday March 13th 2011 at 11:59 PM on `http://cms.csuglab.cornell.edu`.

General Note: This assignment (and future ones) require you to have access to a Unix-like (Linux, Mac OS X, etc) machine. If you do not have such an operating system installed on your local machine, make sure to get a CSUG Lab account. Different systems have slightly different configurations. The main environment in this class will be GNU/Linux.

Assignment Notes:

- You must complete the assignment using GNU/Linux tools that were discussed in class.
- Complete the assignment by submitting your script on CMS.
  - plus a simple feedback on the assignment

Your objective in this assignment is to write a pythoon script, `reader.py` that is a rudimentary "plain-text eBook reader". The idea here is that a user would launch this program, read a little bit of a book and close it. Next time the user launches this program with that book, he continues from where he left off the last time.

The script takes at least one command-line argument, the name of the file to open. The user can also supply an optional paging argument identified by the flag `-n` that dictates how many lines to display per page. If the user did not supply that argument, the script should assume a default paging of 40 lines per page. Finally, for navigation, we will only support one operation: "next page" to display the next page. The next page is displayed when the user presses the *Enter/Return* key.

To make things simple, here is how we're going to make the script work:

1. When the script starts, it checks to see if the directory of the passed-in eBook contains a `.reader` file *(the filename starts with a dot)*. We will use this file to store the progress of different files that directory. So if that directory does not contain a `.reader` file, we will create one.
2. The contents of a `.reader` file will be a two comma-separated columns of `file_name,line_count`. Where the line count is the number of lines the user has read so far. Here are the contents of a sample `.reader` file:

```
moby_dick.txt,120
```

```
art_of_war.txt,80
the_republic.txt,240
```

3. When the program launches, it will open the passed-in file name, read the `.reader` file, and skip the indicated number of lines. Then it will display a page's worth of lines (as dictated by the paging argument) to the user.

4. When the user presses the return key, the script will update the files read line count, rewrite `.reader` file to disk, and print the next page to the terminal.

5. For now we will assume that the user forces the program to quit by pressing `CTRL-C`. So you do not need to implement any special mechanisms to handle exiting.

Summary/Notes:

- Files begining with a dot are hidden in Linux, so you won't see them when you do an `ls`. To see all files, do: `ls -a`
- Just to be clear, here are how you call the program:
  - `./reader.py the_republic.txt` will launch the script with default paging of 40 lines per page.
  - `./reader.py -n 80 the_republic.txt` will launch the script with paging set to 80 lines per page.
- Since a user might use different paging even with the same file, make sure that the `.reader` file contains number of lines displayed and not the number of pages displayed.

Remember your two best friends are your favorite search engine, and the Python documentation: `http://www.python.org/doc/`

Good Luck!