Homework 3

Software Design
Fall 2004

Allen B. Downey

Due: Thursday 23 September.
The reading for this assignment is Chapters 7–8 of *How to think....*

3.1 Palindromes, Lipograms and Tautonyms

1. Download `filters.py` by typing:

   ```
   wget http://wb/sd/code/filters.py
   ```

   Read the file and get an idea of what it does. You should be able to get a sense of how it works even though it uses some features we haven't worked with yet.

   The function `is_palindrome` is an example of a filter. It checks words and indicates whether or not they are palindromes. Or, at least, it is supposed to. At the moment it indicates whether or not a word is equal to the string `'palindrome'`, which is probably not what we had in mind.

   If you run the program as is, it will run through a list of words (in `/usr/share/dict/words`) and print `palindrome`. Your job is to change `is_palindrome` so that it does what it is supposed to do. A palindrome, since you asked, is a word that reads the same forward and backward, like "reviver" and "noon."

   How many palindromes are there in our little dictionary? Hint: use a pipe.

2. Write a filter function named `is_word` that always returns `True`. Now you can run `filters.py` again with the new filter:

   ```
   python filters.py is_word | wc
   ```

   So now we know how many words there are in the dictionary.

3. Read the handout regarding lipograms (attached), and then write a filter called `has_no_e` that returns `True` if the given word doesn't have the letter "e" in it. What percentage of the words in the dictionary have no "e"?

4. Write a function named `avoids` that takes a word and a string of forbidden letters, and that returns true if the word doesn't use any of the forbidden letters. You can call your function from the command line like this:

   ```
   python filters.py avoids xyz | wc
   ```

   Can you find a combination of 5 forbidden letters that excludes the smallest fraction of the dictionary?

5. Write a function that takes a word and a list of required letters, and that returns `True` if the word uses all the required letters at least once. How many words are there that use all the vowels `aeiou`? How about `aeiouy`?

   Hint: You can convert a string to a list using the built-in function `list`.

6. Write a function that takes a word and a list of letters, and that returns `True` if the word contains only letters in the list. How many words can be formed using only the letters `acefhlo`? Other than "hoe alfalfa".

7. Write a function called `is_abecedarian` that returns `True` if the letters in a word appear in alphabetical order. How many abecedarian words are there?

3.2 Gooey

1. Download `Palindrome.py` from the class web page. Read through the program and then run it. If the version of `is_palindrome` in your `filters.py` is correct, then `Palindrome.py` should provide a working graphical palindrome-checker.

2. In `filters.py`, write a function called `rotate_word` that takes a string and an integer as parameters, and that returns a new string that contains the letters from the original string "rotated" by the given amount. To rotate a letter means to shift it through the alphabet, wrapping around to the beginning if necessary. For example, Y shifted by 1 is Z, and Z shifted by 1 is A. A shifted by 5 is F.

   For you hard-drinking electrical engineers, what do you get if you rotate 'ohms' by 1? For you entrepreneurial bakers, what do you get if you rotate 'hawk' by 8? And for you environmentalists, what's 'terra' rotated by 13?

   Hint: check out the `ord` and `chr` functions documented at

   `http://www.python.org/doc/current/lib/built-in-funcs.html`

   A good solution to this problem should be appropriately general and encapsulated. It should work for upper- and lower-case letters, and for positive and negative shifts.

3. As a JFFE, how many rotate-pairs can you find in the dictionary?

4. As a second JFFE, go back to `Palindrome.py` and make a copy of `check_palindrome` called `rotate_entry`. Instead of checking for palindromes, it should take the contents of the text entry, rotate it by 1, and display the result in the label.

5. As a third JFFE, add a second text entry to the GUI, and modify `rotate_entry` so that it gets the amount of the shift from the new entry.

WHAT TO TURN IN: You should hand in a copy of `filters.py` (printed with `a2ps`), along with a page of plain text that presents your findings regarding words with various properties, rotate-pairs, and anything else you find that you think is interesting.