Homework 7

Introductory Programming
Fall 2004

Allen B. Downey

The reading for this assignment is Chapters 8–10 of *How to think....*

7.1 Be fruitful!

1. Create a file named `analyze.py` and type in the following code:

```
from string import *

s = 'Allen!'
s = s.lower()
s = s.strip(whitespace+punctuation)
print s
```

   Read the documentation of the string class and make sure you understand what this code does. Then rewrite it as a function named `clean` that takes a string as a parameter and that returns a new, cleaned up string as a return value. Add code that tests your new function.

2. Add the following lines to your program:

```
def clean_list(words):
    for word in words:
        print clean(word)

line = 'The quick brown fox.'
words = split(line)
clean_list(words)
```

   `clean_list` takes a list of strings, traverses the list, cleans each element, and prints it. The test code creates a string, uses `string.split` to break it into a list of words, and then invokes `clean_list`. Again, make sure you understand this code before proceeding.

   Your mission is to transform `clean_list` into a fruitful function. In other words, instead of cleaning the elements and printing them, it should form and return a new list of strings. Hint: use the `append` method to add new elements to a list.

   As usual, you should write a few lines of code to test your new function.

3. Download the file `gatsby.txt` from the class web page:

```
wget http://wb/ip/code/gatsby.txt
```

   Add the following lines of code to your program:

```
fp = open('gatsby.txt')
for line in fp:
    print line
```

Make sure you understand this code (you might want to read the first two sections of Chapter 11), and then run it. You should see the text of *The Great Gatsby*, by F. Scott Fitzgerald. It might take a while to print; you can hit Control-C to stop it.

Encapsulate this code in a function named `process_file` and test it again. Then modify it so that instead of printing each line, it splits the line and uses `clean_list` to remove punctuation and convert to lower case. For debugging, you should print each cleaned list, but once you have it working, remove the print statement. Then you should count the total number of words in the file by adding up the length of the word lists.

Finally, make `process_file` fruitful; it should return the total number of words in the file. When you are done, your program should read the file and print just the total number of words. Use `wc` to confirm that the total is correct.

Next time, we will extend this program to count the number of `unique` words in the book.