Homework 4

Due: November 14, 2019 **Points:** 100

1. (*50 points*) A *web crawler* is a program that starts at a web site. It then follows the links on that page recursively. If you think of the links as branches of a tree, and the original web site as the root, the crawler follows the branches to the next "root" (i.e., web site) and then repeats this process some number of times (the *depth*).

   Your job is to write a simple web crawler. We will focus only on one type of link, that which begins with "http:".

   Write a program that reads in a URL and the depth as an integer. The program should print the integer and the links on the page.

   All links will look like this: `href="link's url"` so you have to extract the "link's url" part. You can do this in two ways. First, you can use the string method `find()` to look for "`href="`", and then for the closing "`"`", and extract the string between the two. Or, you can use the regular expression package. To do this, import "re" and then use the line:

   `re.findall('href="(http://.*?)"', webpagetextstring)`

   where *webpagetextstring* is the contents of the web page you are checking. This returns a list of links, for example

   ```
   [ 'http://nob.cs.ucdavis.edu/mhi289i/sub1/index.html',
   'http://nob.cs.ucdavis.edu/secure-exer/index.html',
   'http://nob.cs.ucdavis.edu/mhi289i/sub2/index.html',
   'http://nob.cs.ucdavis.edu/mhi289i/next.html' ]
   ```

   Print the links in the following form (your listing may differ):

   ```
   http://nob.cs.ucdavis.edu/mhi289i/index.html contains:
        http://nob.cs.ucdavis.edu/mhi289i/sub1/index.html
        http://nob.cs.ucdavis.edu/secure-exer/index.html
        http://nob.cs.ucdavis.edu/mhi289i/sub2/index.html
        http://nob.cs.ucdavis.edu/mhi289i/next.html
   ```

   or, if there are no links, print:

   ```
   http://nob.cs.ucdavis.edu/secure-exer/index.html contains no links
   ```

   To turn in: Please call your program "crawler1.py" , and submit it to Canvas.

2. (*50 points*) Extend your program in problem 1. Recursively visit each link and print the links on the page, as above. Do this to the indicated depth. Be sure you don't repeat pages; once you print the links, do not print them again should you revisit the page.

   A good web site to test your program on is http://nob.cs.ucdavis.edu/mhi289i/index.html.

   Print the links in the following form (your listing may differ):

   ```
   http://nob.cs.ucdavis.edu/mhi289i/index.html contains:
        http://nob.cs.ucdavis.edu/mhi289i/sub1/index.html
        http://nob.cs.ucdavis.edu/secure-exer/index.html
        http://nob.cs.ucdavis.edu/mhi289i/sub2/index.html
        http://nob.cs.ucdavis.edu/mhi289i/next.html
   http://nob.cs.ucdavis.edu/mhi289i/sub1/index.html contains:
        http://nob.cs.ucdavis.edu/mhi289i/sub2/index.html
        http://nob.cs.ucdavis.edu/mhi289i/index.html
   ```

   Hint: Use a dictionary for this. The key would be the URL of the current web page and the value would be the list of links. Then, when you visit a web page, check that its URL is not in the dictionary. If it is, you already visited it and all its links, so simply return.

   To turn in: Call this program "crawler2.py" when you submit it.

Extra Credit

E1. (*50 points*) This problem asks you to calculate the atomic weight of molecules.

The file "atomic_weights.txt" contains lines with three fields separated by tabs. The first field is the atomic weight, the second field is the symbol for the element, and the third field is the name of the element (which you can ignore for this problem).

We will proceed in stages, to make life easier. You should turn in only the final program, though.

(a) Write a function to load the contents of the file into a dictionary. The key is to be the chemical symbol. The value is to be the atomic weight. As noted above, you can ignore the third field.

(b) Now write a function that takes a chemical compound and breaks it into elements and numbers. The basic unit of a chemical formula is an element's symbol followed by a number (1 or more digits); the chemical compound's formula is a sequence of one or more units. For example, the chemical formula for ethanol, `C2H5OH`, is 2 C (carbon) atoms, 5 H (hydrogen) atoms, an oxygen atom, and another hydrogen atom; and the chemical formula for water, `H2O`, is 2 H (hydrogen) atoms and an oxygen atom.

A good way to check your program is to have it print out each atom's symbol and the number that follows it, if any.

Hint: Element symbols are either 1 or 2 letters. The first letter is *always* capitalized; if there is a second letter, it is *always* lower case. So "HO" is a hydrogen atom (H) and an oxygen atom (O), and "Ho" is the symbol for holmium. Similarly, "SN" is a sulfur atom (S) and a nitrogen atom (N), and "Sn" is the symbol for tin. Similarly, if no number follows an element's name, treat it as 1.

(c) Using the functions you wrote in the above two parts, write a program that reads in a chemical compound and prints its atomic weight. Your program is to continue reading input until the user types an end of file.

Your output is to look like this (input is in red).

```
Chemical composition? C2H5OH↵
The atomic weight of C2H5OH is 46.08
Chemical composition? H2O↵
The atomic weight of H2O is 18.02
Chemical composition? HO↵
The atomic weight of HO is 17.01
Chemical composition? Ho↵
The atomic weight of Ho is 164.93
Chemical composition? SN3↵
The atomic weight of SN3 is 74.1
Chemical composition? Sn3↵
The atomic weight of Sn3 is 356.13
Chemical composition? control-D
```

To turn in: Please call your program "compound.py" and submit it to Canvas.