Assignment 4, TCSS 480 Winter 2016
Due Friday, Feb. 5, 2016, 9 a.m.

OBJECTIVE

The objective of this assignment is to apply your newly learned Python knowledge to build a slightly larger application.

ASSIGNMENT DESCRIPTION

Given the initial text file with URLs, you are to create a web crawler that searches Web pages for outside / absolute links and processes up to 100 Web pages. You need to generate a csv file containing a map of the URL search space and a word cloud consisting of words used in link names as your output.

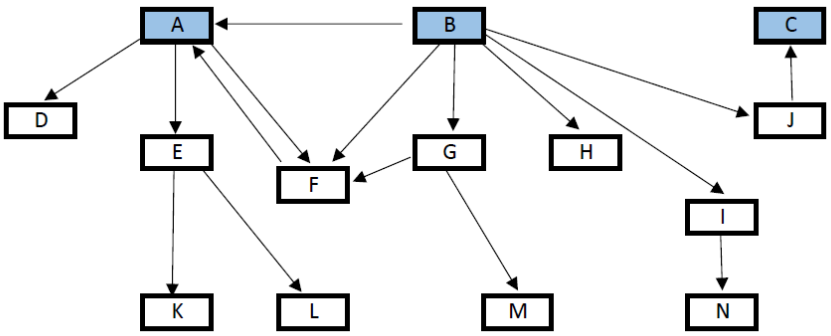
Input: a txt file named `urls.txt` (hardcode the file name) with several URLs (one URL per line). Sample provided – note that the last line of the input file is empty.

Web crawler: for each of the URLs provided in the initial file, find all the absolute links and the names associated with them, and then go through the pages that are linked from the main URLs until you either process 100 pages or exhaust the links. An absolute link defines a specific location of the Web file or document including the protocol, the server name, the directory/s (if any), and the name of the document itself (if any), e.g.
`<a href = "http://www.domain.com/folder1/folder1a/pagename.html">Some text</a>`
In this example, `http://www.domain.com/folder1/folder1a/pagename.html` is the absolute link, whereas `Some text` is the name given to the link.
Assume there are no attributes listed between `a` and `href`. However, remember there may or may not be whitespaces around the = sign. Assume that the actual website address can only begin with `http` or `https`. You are to use regular expressions to find the links in the HTML document.

In order to process webpages, use `urllib.request` module. The `request` module handles opening and reading of URLs. Once the URL is open, you can read the webpage as if you were reading data from a plain text file using a for loop or using `read` and `readline` (returning `bytes` object, not strings), e.g. the code below reads the main page for our Institute of Technology and echo print its HTML to the screen. If needed, you can typecast `bytes` object to a string: in the code below, `pageText` is of type `bytes`, so to store its string version into a variable you may use something like `pt = str(pageText)`

```
import urllib.request
page = urllib.request.urlopen('http://www.tacoma.uw.edu/institute-technology/institute-technology')
pageText = page.read()
print(pageText)
```

The "crawl" space is to be constructed in the breadth-first order. Example:



If the blue nodes are the starting nodes, then the letters denote the order in which a "crawl" space is to be constructed A-Z (first starting nodes, then all of their immediate children, then the children of children unless there is a cycle, etc.).

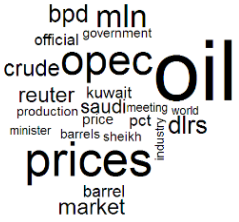
Output: Generate a csv file (25 pts) that shows the "crawl" space as:

Page	Links
A	D, E, F
B	A, F, G, H, I, J
C	
D	
E	K, L
F	A
G	F, M
H	
I	N
J	C
K	
L	
M	
N	

and includes additional information at the bottom that specifies the Web page with the highest number of links pointing to it, in our example F. There may be a possibility that there is a tie for the top spot of course and you need to list all pages that tie for the title of being most popular. Your csv output file should list actual URLs.

In addition, you are to generate a word cloud (15 pts) showing 15 most popular words contained in link text (if < 15 words are present in your cloud, then print them all). You should print the cloud as a dictionary (sorted by the value component) to the console and then in a separate window using graphics. Example:

{'oil' : 10, 'prices': 7, 'opec': 6, 'mln': 4} etc.



SUBMISSION AND GRADING NOTES

You are to submit your finished script through Canvas and the test document only. Name your script `assign4.py`. Your code needs to include comments, use consistent coding style, and has to be compatible with Python 3.4. You are only allowed to use Python built-in modules (i.e. do not use any module that requires additional download). Assignments that do not meet these criteria will receive a grade of 0. If you decide to provide a test document, upload it as txt or pdf, filename does not matter.

This is an individual assignment – you are NOT allowed to work on it with any other student or share your code with others. This assignment is NOT subject to a peer review process and will be graded by the instructor or a grader.

The code will be graded based on its correctness by running it on instructor/grader designed test cases. Then the code will be looked over for anything that violates good coding practice (e.g. lack of code modularization, the use of read() method). If you want to ensure you are given credit even if you break some of our test cases, provide

thorough comments and a text file that explains your testing strategy. The file should provide a short narrative of how you went about the testing process and give a table of test cases, e.g.

Test case	**Reason for the test**	**Expected Outcome**
give Web address/s	To check what happens when no absolute links exist	The csv file contains only starting Web page addresses
give Web address/s	Testing the word cloud	Cloud dictionary
…	…	…

You will not receive any points for your test plan but it may prove beneficial to you in the grading process.