Names: _____ **100 points total**

CS 2123 Programming Project 2 Fall 2019

Assignment is due at 11:59pm on September 26. Submit a digital copy of the assignment on Harvey. You may submit a lateness coupon request BEFORE the assignment is due by sending an email to cs2123f19@googlegroups.com with Subject "CS2123 Project Lateness Coupon". All other late work will receive a 10 percentage point deduction per day (including weekends), No late work is accepted beyond five days after the assignment is due.

In this problem you will write code to efficiently crawl webpages, as well as answer questions about the links between webpages. Consider the following example website: `https://secon.utulsa.edu/cs2123/webtraverse/index.html`. This website has only internal links to other pages on the same website, which makes it ideal for testing. You should turn in a copy of the code and output in a .zip file on Harvey. The Python file must be named `traverse.py`, and you must not modify the names of any of the functions or their defined parameters. Any outputs from running the code should be included in a file called `traverse_output.txt`, also included in the zip file. **Both of these files must be included in a directory called Group#, where # is the group you have been assigned**. Write your names and group number as a comment in the first line of code in traverse.py and at the top of the output file.

Starter code is available from `https://secon.utulsa.edu/cs2123/code/traverse_starter.py`. You may use the helper function `getLinks(url,baseurl)`, which fetches a given URL and extracts all links, returning them as a list of URLs.

Note: you must install the Python package BeautifulSoup 4 (`https://www.crummy.com/software/BeautifulSoup/`), which parses HTML code. Install BeautifulSoup using pip, Python's package installer. On Mac / Linux, run the command "sudo python3 -m pip install beautifulsoup4". On Windows, run the command "python3 -m pip install beautifulsoup4" as administrator.

You should create a dictionary object mapping a URL to a list of links contained in that URL. For example:

```
G['http://secon.utulsa.edu/cs2123/webtraverse/index.html']=
['http://secon.utulsa.edu/cs2123/webtraverse/alink.html',
'http://secon.utulsa.edu/cs2123/webtraverse/blink.html',
'http://secon.utulsa.edu/cs2123/webtraverse/clink.html',
'http://secon.utulsa.edu/cs2123/webtraverse/index.html']
```

You are strongly encouraged to write additional functions that can be called from the required functions in the question. Include output for all the function calls from within the `if __name__=="__main__":` block of the starter code.

a. Implement `print_dfs(url)`: print all links reachable from a starting URL `url` in depth-first order.

b. Implement `print_bfs(url)`: print all links reachable from a starting URL `url` in breadth-first order.

c. Implement `find_shortest_path(url1,url2)`: find and return the shortest path from `url1` to `url2` if one exists.

d. Implement `find_max_depth(url)`: find and return the URL that is the *greatest distance* from `url`, along with the sequence of links that must be followed to reach the page. For this problem, distance is defined as the minimum number of links that must be followed to reach the page.