CS532S19: Assignment #1

Due on Thursday, January 31, 2019

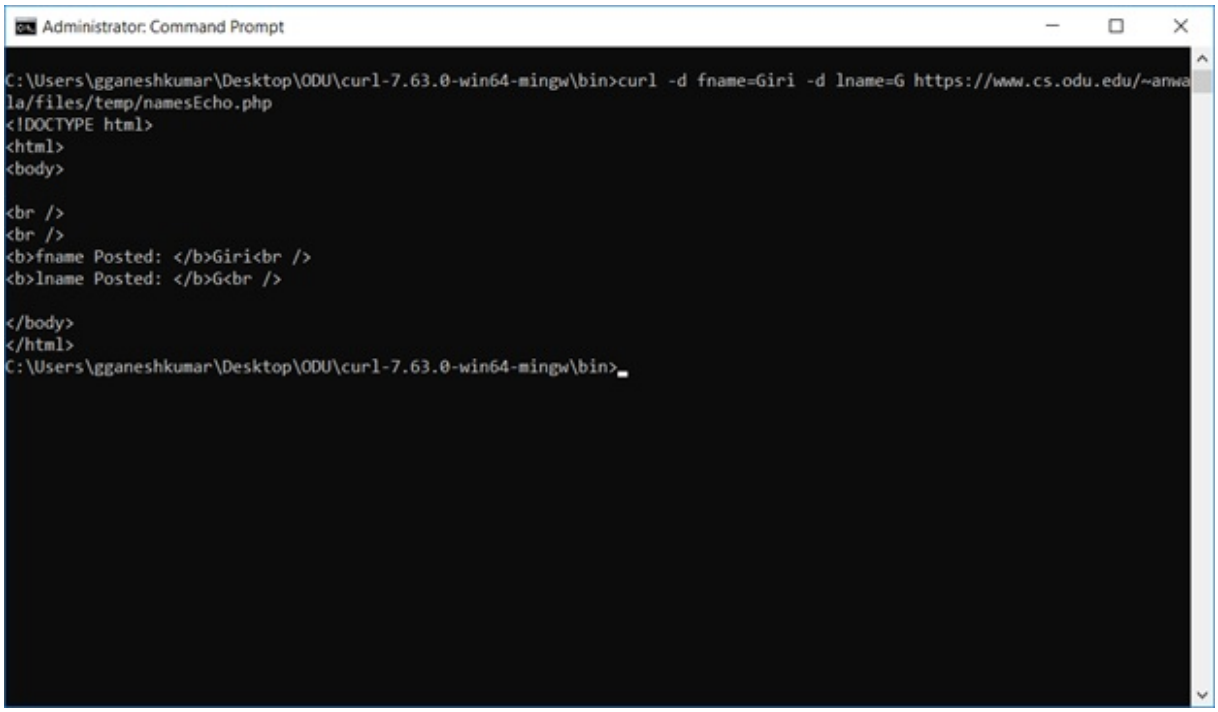
Nwala, Alexander C

Giridharan Ganeshkumar

Question 1

Demonstrate that you know how to use "curl" well enough to correctly POST data to a form. Show that the HTML response that is returned is "correct". That is, the server should take the arguments you POSTed and build a response accordingly. Save the HTML response to a file and then view that file in a browser and take a screen shot

1. As shown in the below command prompt screen shot, a POST request is made to the URI "http://www.cs.odu.edu/ anwala/files/temp/namesEcho.php" through curl.
2. The parameters fname and lname are posted with the attribute/keword d.
3. The response from the server is the html with posted fname and lname in the induvidual b tags respectively.



Question 2

How much wood would a woodchuck chuck if a woodchuck could chuck wood?Write a Python program that:

1. **takes as a command line argument a web page**
2. **extracts all the links from the page**
3. **lists all the links that result in PDF files, and prints outthe bytes for each of the links. (note: be sure to follow all the redirects until the link terminates with a "200 OK".)**
4. **show that the program works on 3 different URIs, one of which needs to be: http://www.cs.odu.edu/ mln/teaching/cs17/test/pdfs.html**

Question 2 continued on next page. . .

1. Use the input to get the URL from the user and store it in a variable
2. To extracts all the links from the page pass the URI to urlib package adn invoke the request.urlopen mentod which returns the response.
3. To parse the response we use the BeautifulSoup which has method called find all which takes the argument as the element we are trying to find. In our case this would be the anchor tag. This might return a few relative URIs as well.
4. As mentioned in line 6 we get the domain and the protocol and append it to the relative URIs alone with an if condition.
5. Now we use the requests class to request the heads which also takes a allow redirect=True as an argument which will make sure we follow the redirects until we find a valid response from the server. Then we check the response headers of each URI opened. In our case we are interseted in content-type matching to the application/pdf and if it does we print the content-length.
6. The screen shot below the code block gives the example for the input as http://www.cs.odu.edu/ mln/teaching/cs532-s17/test/pdfs.html

Listing 1: Python Script

```
1  from bs4 import BeautifulSoup
2  import urllib.request
3  import requests
4  from urllib.parse import urlparse
5  user_Input_Uri = input('Please enter a valid URL:')
6  parsed_uri = urlparse(user_Input_Uri)
7  host_Url_To_Append_To_RelativeUrl = '{uri.scheme}://{uri.netloc}/'.format(
       uri=parsed_uri)
8
9  with urllib.request.urlopen(user_Input_Uri) as res:
10     currentHtml = res.read()
11     soup = BeautifulSoup(currentHtml)
12     for links in soup.find_all('a'):
13         currentUrl = links.get('href')
14         if(currentUrl.startswith('/')):
15             currentUrl = host_Url_To_Append_To_RelativeUrl + currentUrl
16         print(currentUrl)
17         resForEachLinks = requests.head(currentUrl, allow_redirects=True)
18         if(resForEachLinks.status_code == 200 and resForEachLinks.headers['
               Content-Type'] =='application/pdf'):
19             print('The size of the PDF in the above link is: ' +
                   resForEachLinks.headers["content-length"] + ' Bytes')
```

Question 2 continued on next page...

jupyter CS532_Assignment_1_Problem1 Last Checkpoint: 7 hours ago (unsaved changes)

File Edit View Insert Cell Kernel Help Trusted Python 3

In [14]:
```
from bs4 import BeautifulSoup
import urllib.request
import requests
from urllib.parse import urlparse
user_Input_Uri = input('Please enter a valid URL:')
parsed_uri = urlparse(user_Input_Uri)
host_Url_To_Append_To_RelativeUrl = '{uri.scheme}://{uri.netloc}/'.format(uri=parsed_uri)

with urllib.request.urlopen(user_Input_Uri) as res:
    currentHtml = res.read()
    soup = BeautifulSoup(currentHtml)
    for links in soup.find_all('a'):
        currentUrl = links.get('href')
        if(currentUrl.startswith('/')):
            currentUrl = host_Url_To_Append_To_RelativeUrl + currentUrl
        print(currentUrl)
        resForEachLinks = requests.head(currentUrl, allow_redirects=True)
        if(resForEachLinks.status_code == 200 and resForEachLinks.headers['Content-Type'] =='application/pdf'):
            print('The size of the PDF in the above link is: ' + resForEachLinks.headers["content-length"] + ' Bytes')
```

```
Please enter a valid URL:http://www.cs.odu.edu/~mln/teaching/cs532-s17/test/pdfs.html
http://twitter.com/webscidl
http://www.dlib.org/dlib/november15/vandesompel/11vandesompel.html
http://arxiv.org/abs/1508.02315
http://arxiv.org/abs/1508.02315
http://www.cs.odu.edu/~mln/pubs/ht-2015/hypertext-2015-temporal-violations.pdf
The size of the PDF in the above link is: 2184076 Bytes
http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-annotations.pdf
The size of the PDF in the above link is: 622981 Bytes
http://arxiv.org/pdf/1512.06195
The size of the PDF in the above link is: 1748959 Bytes
http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-off-topic.pdf
The size of the PDF in the above link is: 4308768 Bytes
http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-stories.pdf
The size of the PDF in the above link is: 1274604 Bytes
http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-profiling.pdf
The size of the PDF in the above link is: 639001 Bytes
http://dx.doi.org/10.1007/s00799-015-0150-6
http://www.cs.odu.edu/~mln/pubs/jcdl-2014/jcdl-2014-brunelle-damage.pdf
The size of the PDF in the above link is: 2205546 Bytes
http://arxiv.org/abs/1506.06279
http://dx.doi.org/10.1007/s00799-015-0155-1
http://bit.ly/1ZDatNK
The size of the PDF in the above link is: 720476 Bytes
http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-mink.pdf
The size of the PDF in the above link is: 1254605 Bytes
http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-arabic-sites.pdf
The size of the PDF in the above link is: 709420 Bytes
http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-dictionary.pdf
The size of the PDF in the above link is: 2350603 Bytes
http://bit.ly/jcdl-pdf
http://dx.doi.org/10.1007/s00799-015-0140-8
```

Question 3

Consider the "bow-tie" graph in the Broder et al. paper:http://snap.stanford.edu/class/cs224w-readings/broder00bowti
Now consider the following graph:

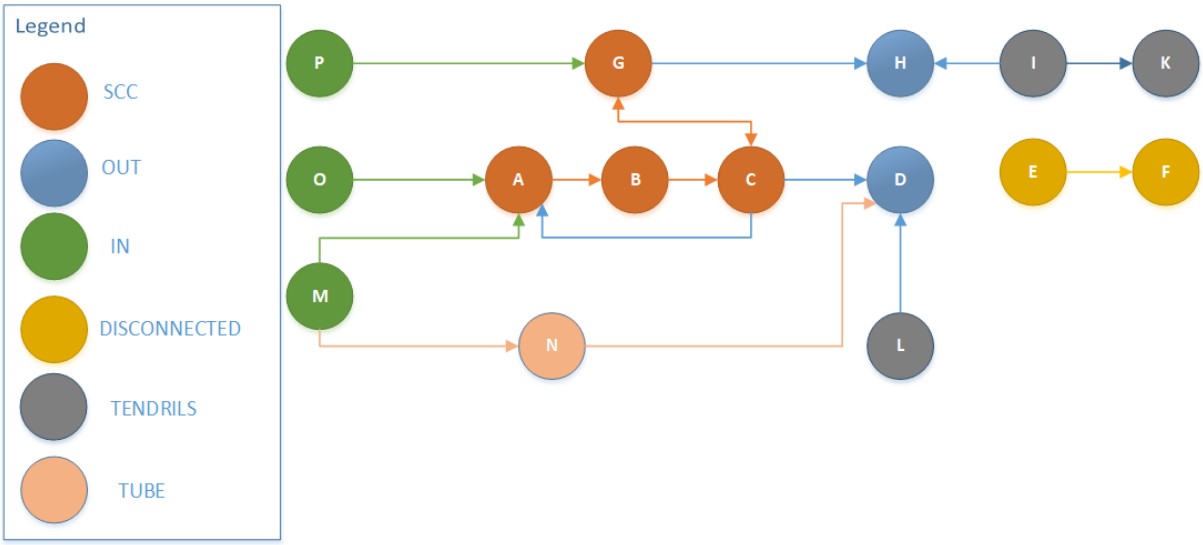
1. **A –> B**
2. **B –> C**
3. **C –> D**
4. **C –> A**
5. **C –> G**
6. **E –> F**
7. **G –> C**
8. **G –> H**
9. **I –> H**
10. **I –> K**
11. **L –> D**
12. **M –> A**
13. **M –> N**
14. **N –> D**

Question 3 continued on next page. . .

15. **O –> A**
16. **P –> G**

For the above graph, give the values for:

1. **IN:**
2. **SCC:**
3. **OUT:**
4. **Tendrils:**
5. **Tubes:**
6. **Disconnected:**



1. SCC: The stongly conencted components as shown in the above figure are all connected to one other strongly, in other words we can reach from all nodes to every other node. All the four nodes are connected in such a eay that we can go from any node to another node.
2. IN: The IN links as shown P, O, M are all nodes which has no in-links but has out links which connect to SCC and Tubes in this case.
3. OUT: H, D in our case as these nodes has no outlinks but has in-links but no out-links.
4. Tendrils: The L node is one the Tendrill but there exists a node J as discussed in the class connected to I node then this along with K node can be considered as Tendrils.
5. Tubes: Tubes are the one that connects the nodes from IN and OUT but would not be part SCC.
6. Disconnected: E and F in our case as these are not connected to any conponents mentioned above.