

CS 595: Assignment 1

Mallika Kogatam

Fall 2014

Contents

1	Problem 1	2
	1.1 Solution	2
	1.2 Code Listing	2
	1.3 Results	4
2	Problem 2	5
	2.1 Solution	5
	2.2 Results	5
	2.3 Code Listing	6
3	Problem 3	8
	3.1 Solution	8
	3.2 Results	9
	3.3 Explanation	9

1 Problem 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.

1.1 Solution

The following steps were taken to become familiar with curl:

1. Executed the simple curl command for couple of websites in order get the source code of that particular page and learned how to inspect the POST method in form tags.
2. Created simple php login page where there are 2 text fields in a form tag, the data entered in the text fields are submitted to the next page through POST method and displayed there.
3. The text fields names are "uname" and "password"
4. With the options -d in curl we can send the arguments in a POST request to the HTTP server, like a browser does when a user had filled in an HTML form and presses the submit button.
5. Option -F in curl is similar to -d option, in addition enables uploading of binary files.

1.2 Code Listing

Here are the two PHP pages to which the curl command is used

loginform.php

```
<html>
  <head>
    <title>Sample</title>
  </head>
  <body>
    <form method="post" action="login.php" id="login_form">
      <label class="hidden-label" for="Email">Username</label>
      <input id="Email" name="u_name" type="text" placeholder="
        Username" value="" spellcheck="false" required />
      <br><br>
      <label class="hidden-label" for="Passwd">Password</label>
      <input id="Passwd" name="password" type="password" placeholder
        ="Password" required />
      <input type="hidden" name="hidden" value="login"/>
      <div class="modal-footer">
        <button type="submit" class="btn btn-primary" name="
          submit">Sign In</button>
      </div>
    </form>
  </body>
</html>
```

login.php

```
<?php
$username = htmlspecialchars($_POST['u_name']);
$password = htmlspecialchars($_POST['password']);
?>
<html>
    <head>
        <title>login </title>
    </head>
    <body>
        <p><?php echo ( !isset($_REQUEST["u_name"])?"": $username ); ?></p>
        <p><?php echo ( !isset($_REQUEST["password"])?"":md5($password)); ?></p>
    </body>
</html>
```

Methodology

1. The following command is used to post data through curl

```
curl -F u_name=mallika -F password=mallika1 www.cs.odu.edu/~mkogatam/fall14/cs595/login.php
OR
curl -d u_name=mallika -d password=mallika1 www.cs.odu.edu/~mkogatam/fall14/cs595/login.php
```

2. Here the argument "mallika" and "mallika1" are posted as user name and password respectively to server through a curl command.
3. When -d option is used in curl it will send the parameters mentioned in the command to a HTTP server through a POST request and gets the output.
4. If the text field name is given wrong then it does not pass any value to the HTTP server.

1.3 Results

Here is the HTML response created from the above command.

```
<html>
  <head>
    <title>login</title>
  </head>

  <body>
    <p>mallika</p>
    <p>0016151e2a3acf1ea17c62a7394eee9e</p>
  </body>
</html>
```

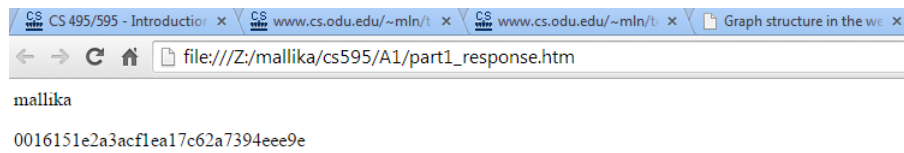


Figure 1: Response for the curl command

2 Problem 2

Write a Python program that:

1. takes one argument, like "Old Dominion" or "Virginia Tech"
2. takes another argument specified in seconds (e.g., "60" for one minute).
3. takes a URI as a third argument:
`http://sports.yahoo.com/college-football/scoreboard/`
or
`http://sports.yahoo.com/college-football/scoreboard/?week=2&conf=all`
or
`http://sports.yahoo.com/college-football/scoreboard/?week=1&conf=72`
etc.
4. dereferences the URI, finds the game corresponding to the team argument, prints out the current score (e.g., "Old Dominion 27, East Carolina 17), sleeps for the specified seconds, and then repeats (until control-C is hit).

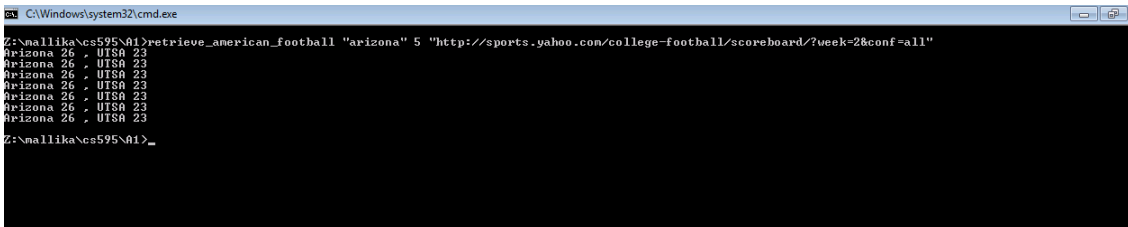
2.1 Solution

1. In order to write this program firstly the source code for the URL should be examined.
2. Find elements in the html code where the team names and scores are located by using "urllib2.urlopen()" function imported from "urllib2" library .check for the div where all the content we want is located and note down the element names
3. Extract the part of code we are looking for by using "BeautifulSoup" library.
4. The 'div' that had score of a teams are read and stored in to a new array
5. Get the Team name as a Argument 1 and sleep time as an Argument 2 from a command line argument.
6. Now looping the stored array, check for the desired team name in the array and get the score associated to the team.
7. By using the sleep method, the program does not exit till the control-C is hit, as the sleep time is given the program keeps updating the result for the given time.

2.2 Results

1. To get the result the below command should be executed

```
retrieve_american_football.py "arizona" 5 \  
"http://sports.yahoo.com/college-football/scoreboard/?week=2&conf=all"
```



```
C:\Windows\system32\cmd.exe  
Z:\nallika\cs595\A1>retrieve_american_football "arizona" 5 "http://sports.yahoo.com/college-football/scoreboard/?week=2&conf=all"  
Arizona 26 , UTSA 23  
Arizona 26 , UTSA 23  
Arizona 26 , UTSA 23  
Arizona 26 , UTSA 23  
Arizona 26 , UTSA 23  
Z:\nallika\cs595\A1>_
```

Figure 2: Sample Output

2.3 Code Listing

```
#!/usr/bin/env python

# Mallika Kogatam
# Fall 2014
# Assignment 1 part 2

import os
import sys
import urllib2
import time

from bs4 import BeautifulSoup

def main():
    if len( sys.argv ) != 4:
        print "Usage_Error"
        sys.exit(0)

    #url = "http://sports.yahoo.com/college-football/scoreboard/"
    url = sys.argv[3]
    team = str( sys.argv[1] ).lower()

    try:
        sleep_time = int( sys.argv[2] )
    except ValueError:
        print "Time_must_be_an_integer"
        sys.exit(1)

    html = urllib2.urlopen( url ).read()
    soup = BeautifulSoup(html)

    while True:
        score_rows = soup.find_all("tbody")[1].find_all("tr")

        for row in score_rows:

            cols = row.find_all("td")

            if len( cols ) == 5:
                cols = cols[1:4]
                #print " "*32

                team1 = cols[0].find_all("em")[0].contents[0]
                team2 = cols[2].find_all("em")[0].contents[0]
                scores = [score.contents[0] for score in cols
                           [1].find_all("span")]

                if team1.lower() == team or team2.lower() ==
                    team:
                    print team1, scores[0], ", ", team2,
                        scores[1]

            time.sleep(sleep_time)
```

```
if __name__ == "__main__":  
    try:  
        main()  
    except KeyboardInterrupt:  
        sys.exit(1)
```


3 Problem 3

Consider the "bow-tie" graph in the Broder et al. paper (fig 9): [urlhttp://www9.org/w9cdrom/160/160.html](http://www9.org/w9cdrom/160/160.html)

Now consider the following graph:

```
A --> B
B --> C
C --> D
C --> A
C --> G
E --> F
G --> C
G --> H
I --> H
I --> J
I --> K
J --> D
L --> D
M --> A
M --> N
N --> D
```

For the above graph, give the values for:

IN:

SCC:

OUT:

Tendrils:

Tubes:

Disconnected:

3.1 Solution

1. First the graph should be drawn in order to figure out which nodes fall under which category.
2. The values for the given graph are:

```
IN: M
SCC: A,B,C,G,
OUT: D,H
In Tendril: None
Out Tendril : L,K,I,J
Tubes: M-->N-->D
Disconnected: E-->F
```

3.2 Results

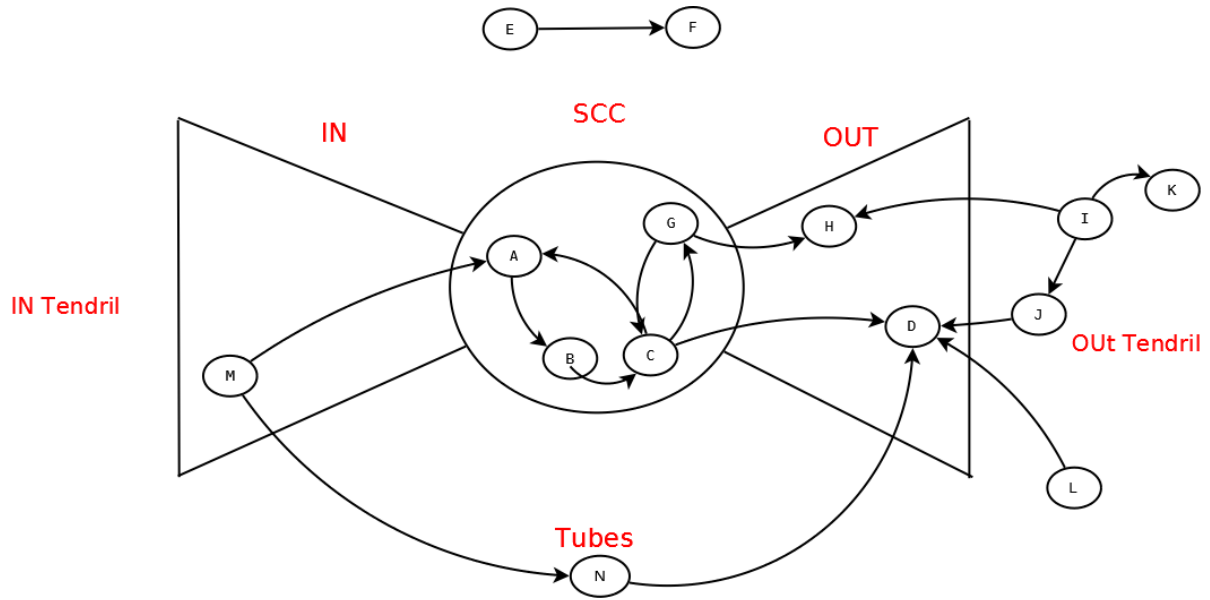


Figure 3: BowTie Graph

3.3 Explanation

1. Its pretty clear that the node E and F are not at all connected to the any part of the graph, they fall under Disconnected.
2. According to the definition of IN,i.e any vertex in IN can be connected to or from any other node in IN, and can be connected to any node in SCC but never from anywhere else, that leaves nodes M in IN.
3. For OUT, the nodes in OUT can be linked from any node in SCC but never the other way. There can never be any node linked from OUT.So that leaves node D,H in OUT.
4. For SCC, the node in scc can be linked from In and linked to OUT. The other feature of SCC is that the nodes in SCC can link to any other node in SCC, that leaves node A,B,C,G.
5. For tubes, the nodes which are linked from IN to Out fall under Tubes. These can not connect to any nodes in SCC, that leaves Nodes N Under Tubes.
6. Tendrils: Tendrils are two types In Tendrils and Out Tendrils. The In Tendrils are the nodes that can never connect to any other nodes in SCC,tubes or OUT
7. Out Tendrils: Out Tendrils are the tendril which can never connect to any other nodes other than the nodes in OUT. That leaves L,K,I,J as Out Tendrils.

Bibliography

- [1] BeautifulSoup documentation. <http://www.crummy.com/software/BeautifulSoup/bs4/doc/>.
- [2] How to fetch internet resources using urllib2. <https://docs.python.org/2/howto/urllib2.html>.
- [3] The web graph. <http://nlp.stanford.edu/IR-book/html/htmledition/the-web-graph-1.html>.
- [4] Farzin Maghoul¹ Andrei Broder¹, Ravi Kumar². Graph strducture in the web. <http://www9.org/w9cdrom/160/160.html>.
- [5] David Easley and Jon Kleinberg. *Networks, Crowds, and Markets: Reasoning about a Highly Connected World*, chapter 13, pages 375–396. Cambridge University Press, 2010.