

ASSIGNMENT # 1

SAI CHAITANYA TIRUMERLA

1. Posting data to a form using curl :

curl is a client to get documents/files from or send documents to a server, using any of the supported protocols (HTTP, HTTPS, FTP, GOPHER, DICT, TELNET, LDAP or FILE). The command is designed to work without user interaction or any kind of interactivity.[1]

To post the data to the url curl takes an argument “--data” (long argument) , shortly “-d” . Posting the data involves either performing logins or to just post the data to a normal form or both. Posting to a data is performed by searching the source for the appropriate parameters of the form example name of the each value then passing necessary values ,the values which we wanted to post to the parameters does the trick. The command used to post the data to a form would be : [2]

```
curl --data  
"param1=value1&param2=value2"  
"URLname" -o filename
```

-o argument takes the output and stores it in a file which is specified by path filename

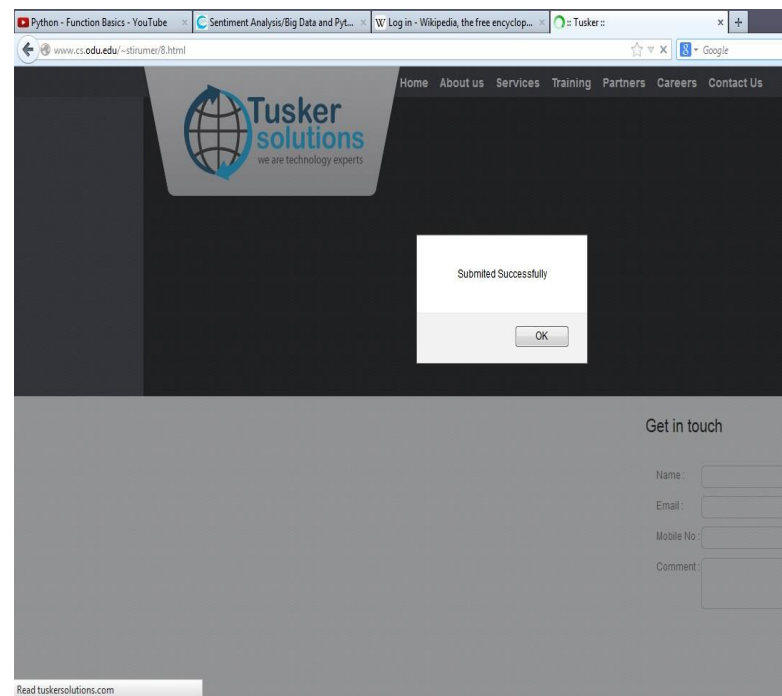
Param1 and param2 are the parameters that are fetched from the source page of the URL and user values value1 and value2 are passed to them respectively.

URL name/path is the URL to which the posting of data is done.

Demo – I have taken sample url to perform the curl operation to check the posting of data and the response retrieved is correct or not.

```
curl --data  
gname=chaitu&gmail=123@gmail.com&gmobile=888-888-8888&gcontent=1213324354kdjkafjks&sub=Submit http://tuskersolutions.com -o  
public_html/8.html
```

Output :



2. Python Script for taking three arguments and displaying the score according to the “URL and Team arguments” :

Pseudocode:

```
#Fetching arguments from user
#Displaying the number of arguments given by user
#Second argument as number of seconds , condition
to display number of minutes according to the
number of seconds passed
#Defining a function to parse the argument i.e. URL
#User's input for the week's number , condition to
check if it between 0 and 3 else display invalid week
#If True , open the url using urlopen function[3]
#Downloads the source code of the url
#Using beautiful soup to parse the HTML and
extracting only the text[4]
#Removing noisy data using indexing and fetching
the required text
#Format according to the text and store it in a list[5]
#Formatting according to the output like the team
score and team with which it played[5]
#Searching the list with the user provided argument
team name
#Fetch the result
#Remove invalid data
#Display the output
#Refresh the page every 30 seconds until user has
terminated.
```

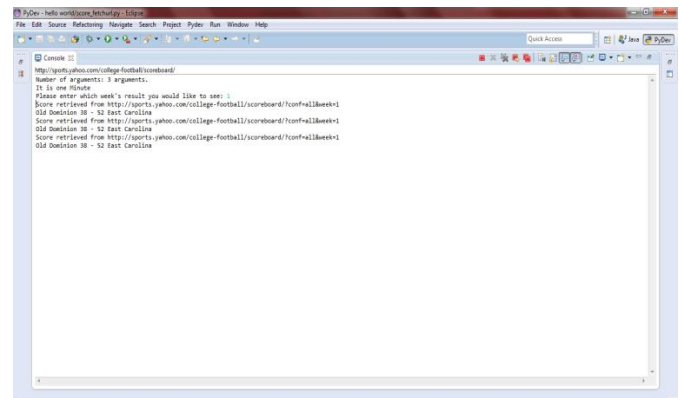
Packages used : sys, beautiful soup , urllib
and time

Url Used : <http://sports.yahoo.com/college-football/scoreboard/>

Script command –
score_fetchurl.py “Old Dominion” 60
<http://sports.yahoo.com/college-football/scoreboard/>

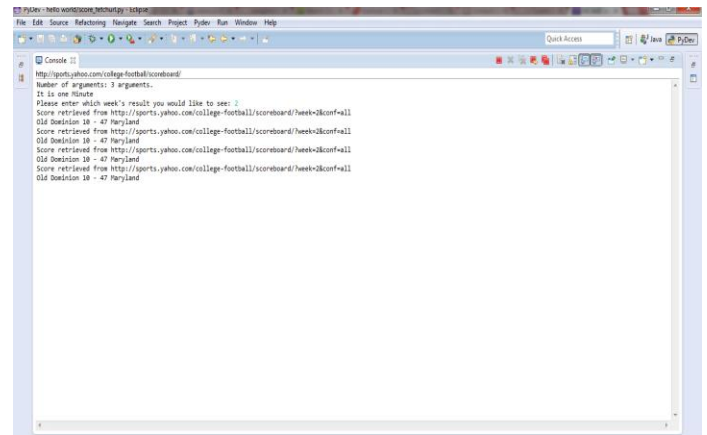
output –

Testing for Week 1:



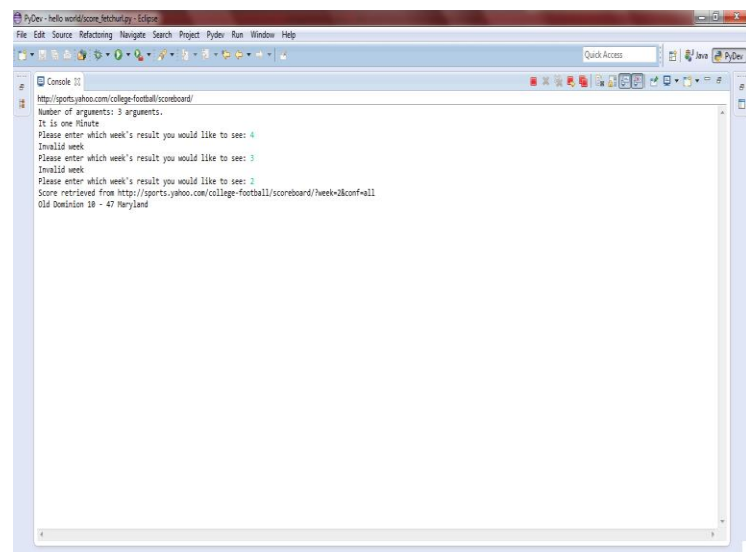
```
File Edit Source Refactoring Navigate Search Project PyDev Run Window Help
PyDev - hello world\score_fetchurl.py - Eclipse
Console [1]
http://sports.yahoo.com/college-football/scoreboard/
Number of arguments: 3 arguments.
It is one Minute
Please enter which week's result you would like to see: 1
Score retrieved from http://sports.yahoo.com/college-football/scoreboard/?conf=all&week=1
Old Dominion 18 - 52 East Carolina
Score retrieved from http://sports.yahoo.com/college-football/scoreboard/?conf=all&week=1
Old Dominion 18 - 52 East Carolina
Score retrieved from http://sports.yahoo.com/college-football/scoreboard/?conf=all&week=1
Old Dominion 18 - 52 East Carolina
```

Testing for week 2:



```
File Edit Source Refactoring Navigate Search Project PyDev Run Window Help
PyDev - hello world\score_fetchurl.py - Eclipse
Console [1]
http://sports.yahoo.com/college-football/scoreboard/
Number of arguments: 3 arguments.
It is one Minute
Please enter which week's result you would like to see: 2
Score retrieved from http://sports.yahoo.com/college-football/scoreboard/?week=2&conf=all
Old Dominion 18 - 47 Maryland
Score retrieved from http://sports.yahoo.com/college-football/scoreboard/?week=2&conf=all
Old Dominion 18 - 47 Maryland
Score retrieved from http://sports.yahoo.com/college-football/scoreboard/?week=2&conf=all
Old Dominion 18 - 47 Maryland
Score retrieved from http://sports.yahoo.com/college-football/scoreboard/?week=2&conf=all
Old Dominion 18 - 47 Maryland
```

Testing for Week 3,4:< Invalid week>



```
File Edit Source Refactoring Navigate Search Project PyDev Run Window Help
PyDev - hello world\score_fetchurl.py - Eclipse
Console [1]
http://sports.yahoo.com/college-football/scoreboard/
Number of arguments: 3 arguments.
It is one Minute
Please enter which week's result you would like to see: 4
Invalid week
Please enter which week's result you would like to see: 3
Invalid week
Please enter which week's result you would like to see: 3
Score retrieved from http://sports.yahoo.com/college-football/scoreboard/?week=3&conf=all
Old Dominion 18 - 47 Maryland
```

<http://www9.org/w9cdrom/160/160.html>

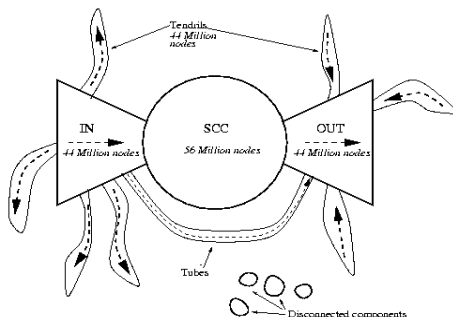
3.

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:

To calculate the values of the above graph which is in the form of a directed graph $G=(V,E)$ where V are the vertices and E are the edges . To compute the values of bow-tie graph which is part of a web graph was given in Broder et al. paper (fig 9)[7]



The main component in the graph is SCC as it is the “CORE” and rest of the values can be found using SCC.

So to calculate SCC several steps are followed which are suggested in [2]:

1 . Compute DFS for Graph G to calculate the fastest time for the nodes

DFS_G

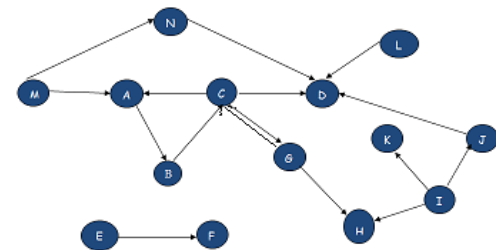
2. Reverse the arcs of the graph and compute DFS from the increasing order of fastest times on the graph

$DFS(G^T)$

3. Then SCC is given by intersection of DFS on graph G and DFS on graph G^T

$$SCC(S) = DFS(G) \cap DFS(G^T)$$

Computing DFS on Graph G ,



With the DFS the fastest times are calculated for each node .

Now reversing the arcs to compute the DFS on G^T from the descending order of fastest times.

Let $G = (V, A)$ be a digraph and let S be a strongly connected component of G . The bow-tie decomposition of G with respect to S consists of the following sets of nodes: [8]

$$\begin{aligned} \text{SCC} &= S = \{\mathbf{A}, \mathbf{B}, \mathbf{C}\}, \{\mathbf{G}\} \\ \text{IN} &= \{v \in V - S \mid S \text{ is reachable from } v\} = \{\mathbf{M}\} \\ \text{OUT} &= \{v \in V - S \mid v \text{ is reachable from } S\} = \{\mathbf{D}\}, \{\mathbf{H}\} \\ \text{TUBES} &= \{v \in V - S - \text{IN} - \text{OUT} \mid \\ &v \text{ is reachable from IN and} \\ &\text{OUT is reachable from } v\} = \{\mathbf{N}\} \\ \text{INTENDRILS} &= \{v \in V - S \mid \\ &v \text{ is reachable from IN and} \\ &\text{OUT is not reachable from } v\} = \{\mathbf{None}\} \\ \text{OUTTENDRILS} &= \{v \in V - S \mid \\ &v \text{ is not reachable from IN and} \\ &\text{OUT is reachable from } v\} = \{\mathbf{I}, \mathbf{J}\}, \{\mathbf{L}\} \\ \text{TENDRILS} &= \text{INTENDRILS} + \\ &\text{OUTTENDRILS} = \{\mathbf{I}, \mathbf{J}\}, \{\mathbf{L}\} \\ \text{OTHERS} &= V - S - \text{IN} - \text{OUT} = \{\mathbf{K}\}, \{\mathbf{E}, \mathbf{F}\} \\ &\text{Where } \{\mathbf{E}, \mathbf{F}\} \text{ are DISCONNECTED} \end{aligned}$$

References :

- [1]http://linux.about.com/od/commands//blcmdl1_curl.htm
- [2]<http://stackoverflow.com/questions/9446085/using-curl-to-post-data-to-a-form>
- [3]<http://docs.python.org/2/library/urllib.html>
- [4]<http://www.crummy.com/software/BeautifulSoup/bs4/doc/>
- [5]<http://stackoverflow.com/questions/7760869/list-printing-in-formatted-string-in-python>
- [6]<http://www.python.org/doc/>
- [7]<http://www9.org/w9cdrom/160/160.html>
- [8]http://isif.org/fusion/proceedings/Fusion_2011/data/papers/267.pdf