Ryan Condotta

Final Report Assignment 1

Question 1:

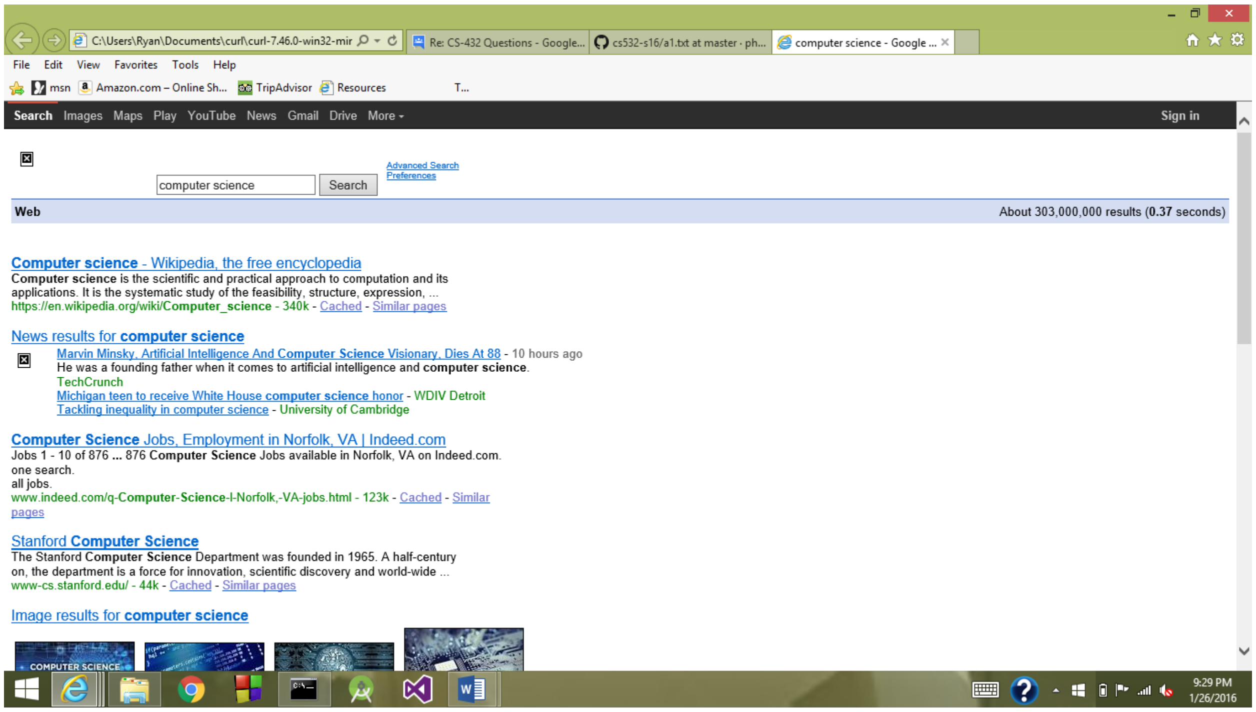
In order to solve the problem of posting data to a form was I had to familiarize myself with curl since it was something that I have never used before. This involved quite a bit of research of the components of the curl which turned out to be successful from my results. The code is shown below:

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Users\Ryan>C:\Users\Ryan\Documents\curl\curl-7.46.0-win32-mingw\bin\curl.exe
-A Mozilla/4.0 http://www.google.com/search?q=computer%20science > C:/Users/Ryan
/Documents/curl/curl-7.46.0-win32-mingw/bin/Assignment1.html

  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 56058    0 56058    0     0  62986      0 --:--:-- --:--:-- --:--:-- 64066
```

The –A Mozilla/4.0 is used to trick the google server in order to accept the command that follows that which is the webpage of http://www.google.com where /search?q=computer%20science will plug in computer science into the search bar while the > C:/Users/Ryan /Documents/curl/curl-7.46.0-win32-mingw/bin/Assignment1.html downloads the resulting information to an html file which loads a web page as shown in Figure 1.



C:\Users\Ryan\Documents\curl\curl-7.46.0-win32-mir
Re: CS-432 Questions - Google...
cs532-s16/a1.txt at master · ph...
computer science - Google ...
File Edit View Favorites Tools Help
msn
Amazon.com – Online Sh...
TripAdvisor
Resources
T...
Search Images Maps Play YouTube News Gmail Drive More
Sign in
Advanced Search
Preferences
9:29 PM
1/26/2016

computer science [Search]

Web About 303,000,000 results (**0.37** seconds)

Computer science - Wikipedia, the free encyclopedia
Computer science is the scientific and practical approach to computation and its applications. It is the systematic study of the feasibility, structure, expression, ...
https://en.wikipedia.org/wiki/**Computer_science** - 340k - Cached - Similar pages

News results for **computer science**
Marvin Minsky, Artificial Intelligence And **Computer Science** Visionary, Dies At 88 - 10 hours ago
He was a founding father when it comes to artificial intelligence and **computer science**.
TechCrunch
Michigan teen to receive White House **computer science** honor - WDIV Detroit
Tackling inequality in computer science - University of Cambridge

Computer Science Jobs, Employment in Norfolk, VA | Indeed.com
Jobs 1 - 10 of 876 **...** 876 **Computer Science** Jobs available in Norfolk, VA on Indeed.com.
one search.
all jobs.
www.indeed.com/q-**Computer-Science**-l-Norfolk,-VA-jobs.html - 123k - Cached - Similar pages

Stanford **Computer Science**
The Stanford **Computer Science** Department was founded in 1965. A half-century on, the department is a force for innovation, scientific discovery and world-wide ...
www-cs.stanford.edu/ - 44k - Cached - Similar pages

Image results for **computer science**

Figure 1

Question 2:

In order to create a Python program that takes a web page, extracts all the links from the page, lists all the PDFs of the links with byte size, and test it on three separate URI's, I had to first study the lecture notes on Python and reteach myself the basic syntax of it. Once the basis was done, I began to tackle each component one at a time beginning with the command line prompt to accept a web page. This was accomplished by taking importing the sys library to take the arguments and store them in the url variable as shown.

```
url = sys.argv[1]
```

The next component I tackled was extracting all the links from the web page and I also listed the extracted links for the user to see. I imported Beautiful Soup and urllib request in order to request all of the href components from the webpage. This is shown from the snippet of code:

```
response = urllib.request.urlopen(url)
soup = BeautifulSoup(response)
count = 0
print("Files Extracted:")
for links in soup.find_all('a'):
        try:
                response1 = urllib.request.urlopen(links.get('href'))
                print(links.get('href'))
                count = count + 1
        except:
                pass
```

The code above works by storing the request of the url in the response variable. I used that variable and plug it into the Beautiful soup to get the html and store it in the soup variable. Then, I used the for loop to find all of the 'a' component. Once all the 'a' components were found I could search for the 'href' to find the links and prints the links. I also put the try and except to take make sure that the program did not crash if an exception was not found.

Once I extracted and listed all of the links that were found from the web page. I had to search for the PDF's link and listed only the PDF links when found with the byte size of each of the links found. The code is shown below:

```
for links in soup.find_all('a'):
        try:
                response1 = urllib.request.urlopen(links.get('href'))
                if response1.info()['Content-type'] == "application/pdf":
                        print(links.get('href'))
                        print("Content-type: ", response1.info()['Content-type'])
                        print("The byte size is: ", response1.info()['Content-Length'])
                        print(" ")
                        pdfcount = pdfcount +1
        except:
                pass
print("Number of PDFs is: " + str(pdfcount))
print("The response code:", response.code)
```

This is the same process for extraction of the links when I listed them. When I get the response from the url with the link reference. I also got other information from the request which made it easier to separate the PDF links with the byte size. I just had to use to response info of content type = "application/pdf" to find which of the links were PDFs and then used Content-Length to find the byte size of the link.

In order to save space on the paper, I have created the document Condotta_Ryan_Assignment1_Q2 which contains the examples for testing on the three separate URI's as well as the workings of the first three parts of the problem.

Question 3:

A bow tie graph was given to me and the paper http://www9.org/w9cdrom/160/160.html described the workings of the components IN, SCC, OUT, Tendrils, Tubes, and Disconnected. The first thing I had to do in order to figure out which node belong to what component was to define each of the components. I noticed from Figure 9 in the paper that each node of IN were connected to the nodes of SCC and SCC was connected to the OUT node. Tendrils are nodes that are connected off of the IN or OUT node and not connected to the SCC node. Tubes are connection between the IN and OUT node and skips over the SCC node. Disconnected are nodes that are not even connected to the graph or completely separated from IN, OUT, Tubes, Tendrils and SCC nodes. Based on this information that was given, I was able to construct the graph as shown in Figure 2 below.

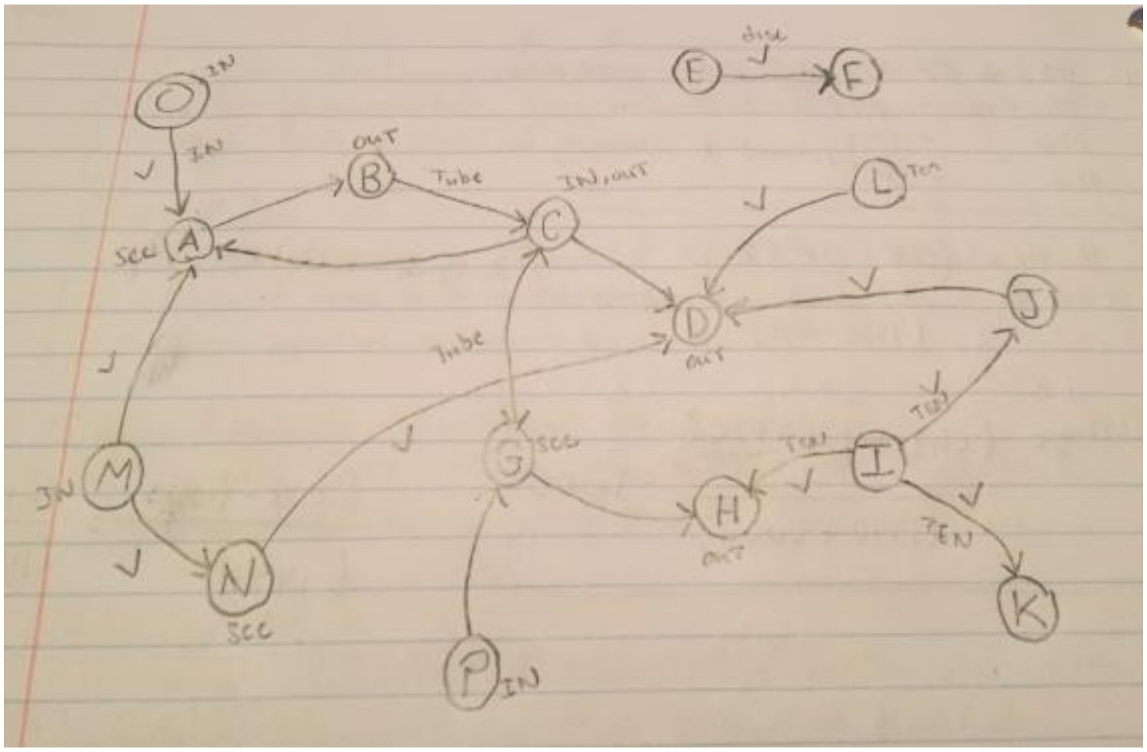


Figure 2

Once the graph was constructed and I knew the individual components, I was able to develop each of the individual components as shown below.

IN: O, M, I, P
SCC: A, G, N, J
OUT: B, C, D, H
Tendrils: (L, D) (I, K)
Tubes: (B, C) (I, H) (C, G)
Disconnected: (E, F)