CS 432/532 Web Science: Assignment #1

Alexander C. Nwala

Bethany DeMerchant

31 January 2019

Contents

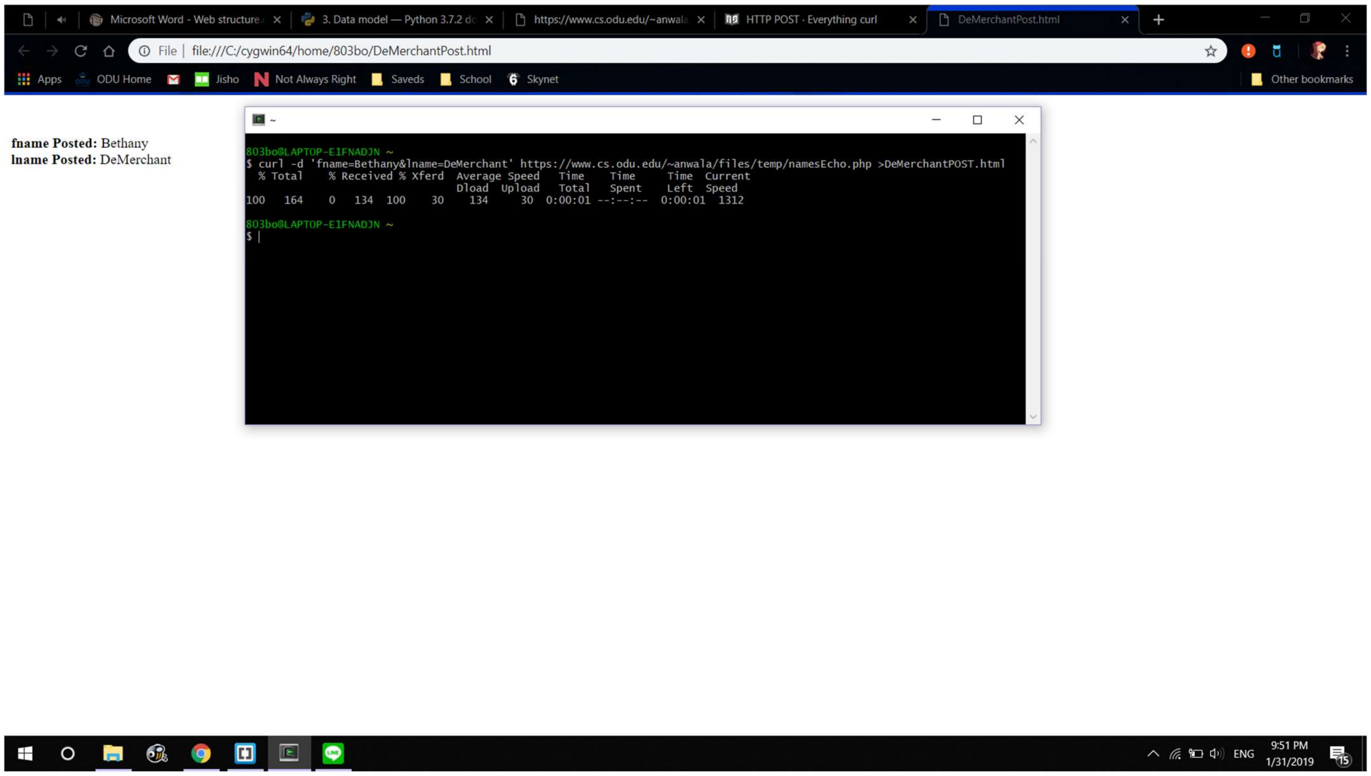
Bethany DeMerchant Web Science Assignment 1

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. Feel free to use my simple server for sending POST requests:
`http://www.cs.odu.edu/~anwala/files/temp/namesEcho.php.`
The server needs you to POST data for "fname" and "lname" fields.

Solution 1:

The solution simply involves using curl with the -d or --data option to post data to a site, as seen in Figure 1. The command in full to post the name Bethany DeMerchant to the given test site is:
`curl -d 'fname=Bethany&lname=DeMerchant'` [`https://www.cs.odu.edu/~anwala/files/temp/namesEcho.php`](https://www.cs.odu.edu/~anwala/files/temp/namesEcho.php)



fname Posted: Bethany
lname Posted: DeMerchant

```
803bo@LAPTOP-E1FNADJN ~
$ curl -d 'fname=Bethany&lname=DeMerchant' https://www.cs.odu.edu/~anwala/files/temp/namesEcho.php >DeMerchantPOST.html
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   164    0   134  100    30    134     30  0:00:01 --:--:--  0:00:01  1312

803bo@LAPTOP-E1FNADJN ~
$ |
```

Figure 1: Post Data Using curl

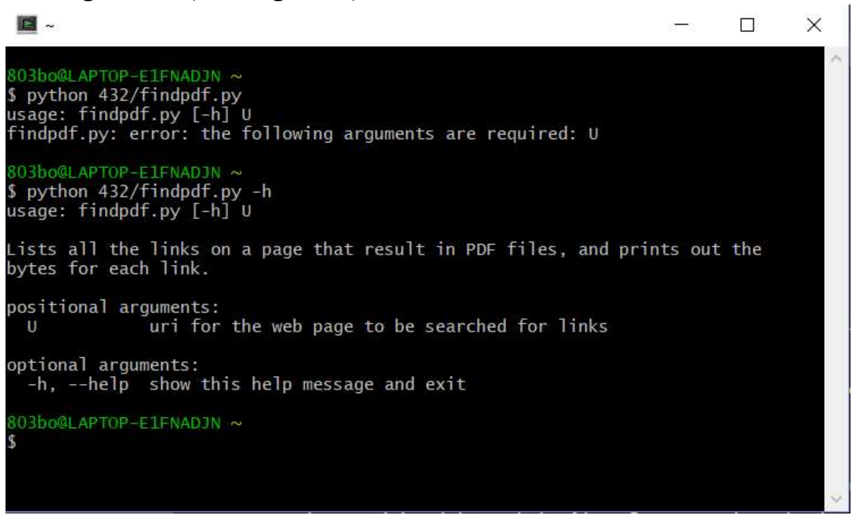
Problem 2:

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 out the bytes for each of the links. (4) Show that the program works on 3 different URIs, one of which needs to be:
`http://www.cs.odu.edu/~mln/teaching/cs532-s17/test/pdfs.html`

Solution 2:

The solution was executed in Python using external modules Requests (http://docs.python-requests.org/en/master/) and BeautifulSoup (https://www.crummy.com/software/BeautifulSoup/) as well as built-in modules argparse and urllib.parse.

The argparse module was used to generate command-line interface and ensure proper functioning if a user fails to provide a URI argument (see Figure 2).

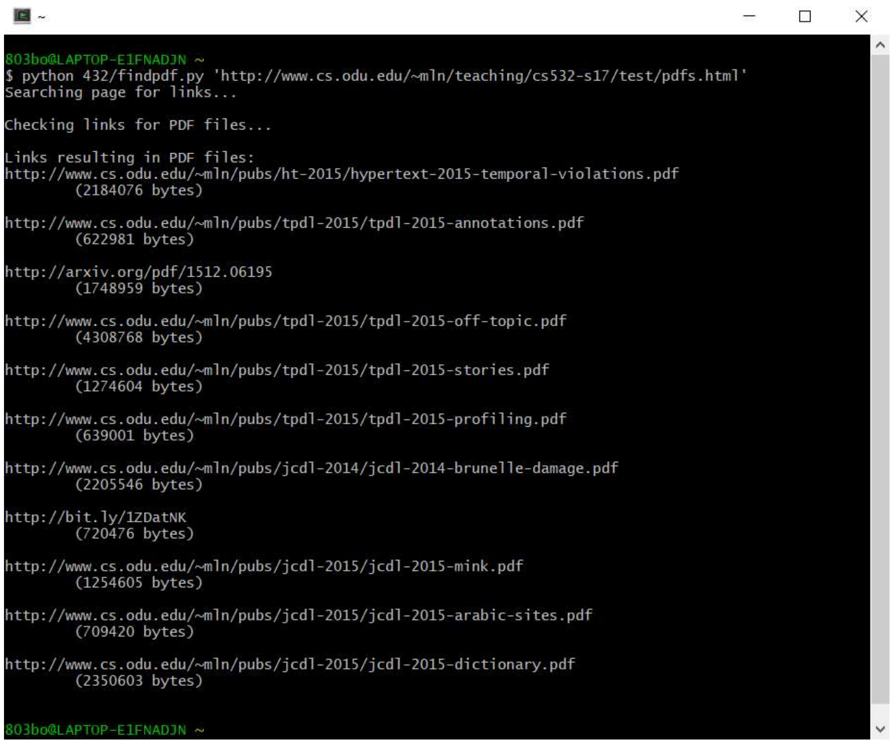


```
803bo@LAPTOP-E1FNADJN ~
$ python 432/findpdf.py
usage: findpdf.py [-h] U
findpdf.py: error: the following arguments are required: U

803bo@LAPTOP-E1FNADJN ~
$ python 432/findpdf.py -h
usage: findpdf.py [-h] U

Lists all the links on a page that result in PDF files, and prints out the
bytes for each link.

positional arguments:
  U           uri for the web page to be searched for links

optional arguments:
  -h, --help  show this help message and exit

803bo@LAPTOP-E1FNADJN ~
$
```

Figure 2: Command-Line Interface Using argparse

The URI provided as an argument is dereferenced using the Requests module, which performs a GET request and creates a response object containing the page contents. Assuming the page contents exist, they are then parsed by BeautifulSoup, which scans for the <a> tag which represents a hyperlink in HTML. For each <a> tag found, the URI address of the link is added to a list of all links in the page. Each address in the complete list is then checked with a HEAD request performed using the Requests module. If the header's Content-Type header is "application/pdf", the page's URI and the value of the Content-Length header are added to a list of links resulting in PDF files. Finally, the finished list of links to PDF files is printed to the console.

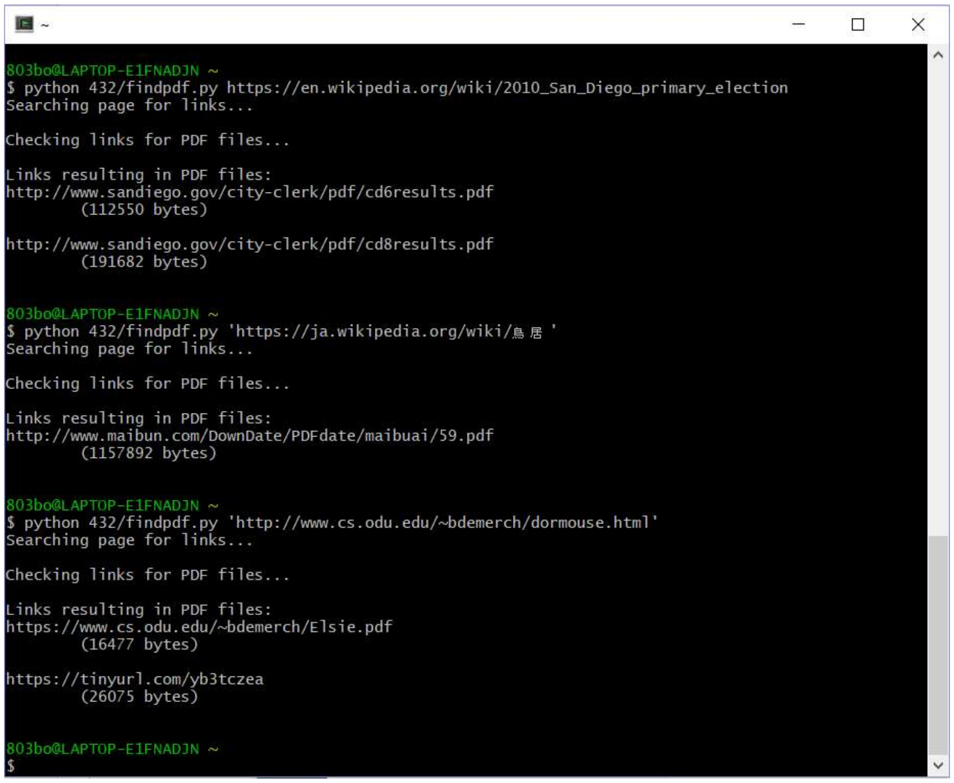
The demonstration page (`http://www.cs.odu.edu/~mln/teaching/cs532-s17/test/pdfs.html`) contains twenty links, eleven of which lead to PDF files. Figure 3 shows that the program reported all eleven PDF files and no links to non-PDF pages, and also that it successfully followed redirects.



```
803bo@LAPTOP-E1FNADJN ~
$ python 432/findpdf.py 'http://www.cs.odu.edu/~mln/teaching/cs532-s17/test/pdfs.html'
Searching page for links...

Checking links for PDF files...

Links resulting in PDF files:
http://www.cs.odu.edu/~mln/pubs/ht-2015/hypertext-2015-temporal-violations.pdf
        (2184076 bytes)

http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-annotations.pdf
        (622981 bytes)

http://arxiv.org/pdf/1512.06195
        (1748959 bytes)

http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-off-topic.pdf
        (4308768 bytes)

http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-stories.pdf
        (1274604 bytes)

http://www.cs.odu.edu/~mln/pubs/tpdl-2015/tpdl-2015-profiling.pdf
        (639001 bytes)

http://www.cs.odu.edu/~mln/pubs/jcdl-2014/jcdl-2014-brunelle-damage.pdf
        (2205546 bytes)

http://bit.ly/1ZDatNK
        (720476 bytes)

http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-mink.pdf
        (1254605 bytes)

http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-arabic-sites.pdf
        (709420 bytes)

http://www.cs.odu.edu/~mln/pubs/jcdl-2015/jcdl-2015-dictionary.pdf
        (2350603 bytes)


803bo@LAPTOP-E1FNADJN ~
```

Figure 3: Results from Demonstration Page

The program was also tested on two Wikipedia articles and a test page created specifically for the purpose. The results can be seen in Figure 4.



```
803bo@LAPTOP-E1FNADJN ~
$ python 432/findpdf.py https://en.wikipedia.org/wiki/2010_San_Diego_primary_election
Searching page for links...

Checking links for PDF files...

Links resulting in PDF files:
http://www.sandiego.gov/city-clerk/pdf/cd6results.pdf
        (112550 bytes)

http://www.sandiego.gov/city-clerk/pdf/cd8results.pdf
        (191682 bytes)


803bo@LAPTOP-E1FNADJN ~
$ python 432/findpdf.py 'https://ja.wikipedia.org/wiki/鳥居'
Searching page for links...

Checking links for PDF files...

Links resulting in PDF files:
http://www.maibun.com/DownDate/PDFdate/maibuai/59.pdf
        (1157892 bytes)


803bo@LAPTOP-E1FNADJN ~
$ python 432/findpdf.py 'http://www.cs.odu.edu/~bdemerch/dormouse.html'
Searching page for links...

Checking links for PDF files...

Links resulting in PDF files:
https://www.cs.odu.edu/~bdemerch/Elsie.pdf
        (16477 bytes)

https://tinyurl.com/yb3tczea
        (26075 bytes)


803bo@LAPTOP-E1FNADJN ~
$
```

Figure 4: Results of Secondary Testing

Problem 3:

Consider the "bow-tie" graph in the Broder et al. paper:
`http://snap.stanford.edu/class/cs224w-readings/broder00bowtie.pdf`
Now consider the following graph[1]:

$$\begin{array}{cccccccccccccccc} A & B & C & C & C & E & G & G & I & I & L & M & M & N & O & P \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ B & C & D & A & G & F & C & H & H & K & D & A & N & D & A & G \end{array}$$

For the above graph, give the values for: IN, SCC, OUT, Tendrils, Tubes, Disconnected.

Solution 3:

The first step to this solution is identifying the connections between the graph nodes (see Figure 5).

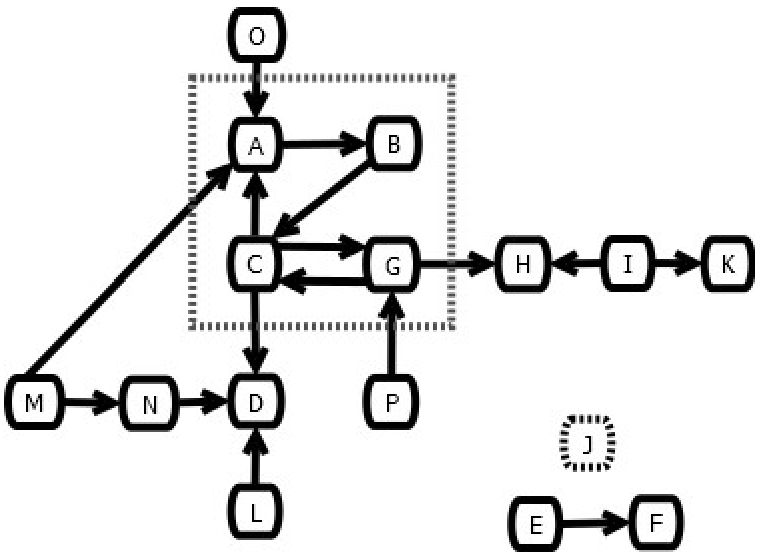


Figure 5: Directed Graph of Connections

The nodes in the dotted square (A, B, C, and G) form the SCC. Each of these nodes can be reached with a path of the other nodes from any other node in the square. Nodes O, M, and P are all IN. They each have no in-links and have out-links to other pages. Nodes H and D are OUT; they have only out-links. Nodes E, F, and J[2] are disconnected. While E and F are connected to each other, none of them are connected to the rest of the network. Node L is a tendril, it has a link only to OUT (D). Node N is a tube; it is linked from IN and links to OUT but cannot reach the SCC. Node I is a tendril; it has only out-links to the other pages. Node K is then OUT, as defined in the secondary guide provided[3]; it has no out-links and its only in-link is from a tendril.

[1] The graph has been reproduced horizontally in an effort to conserve space.
[2] Node J, while not specifically referenced, is assumed to exist due to the existence of nodes I and K-P.
[3] `https://www.harding.edu/fmccown/classes/archive/comp475-s13/web-structure-homework.pdf`