**Assignment Two**

Jacob Berlin

CS432 – Spring 2016

Write a Python program that extracts 1000 unique links from

Twitter. You might want to take a look at:

http://thomassileo.com/blog/2013/01/25/using-twitter-rest-api-v1-dot-1-with-python/

But there are many other similar resources available on the web. Note

that only Twitter API 1.1 is currently available; version 1 code will

no longer work.

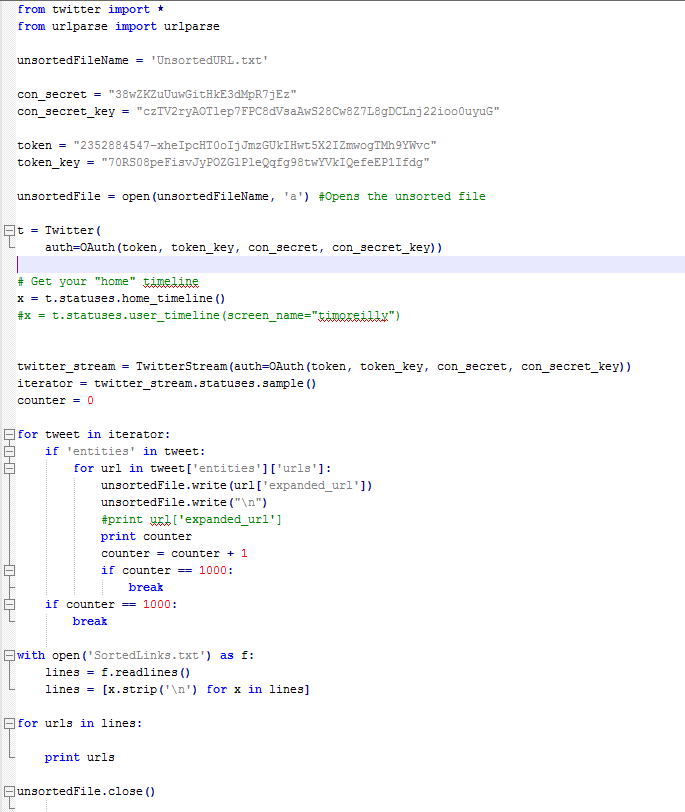
Also note that you need to verify that the final target URI (i.e., the

one that responds with a 200) is unique. You could have many different

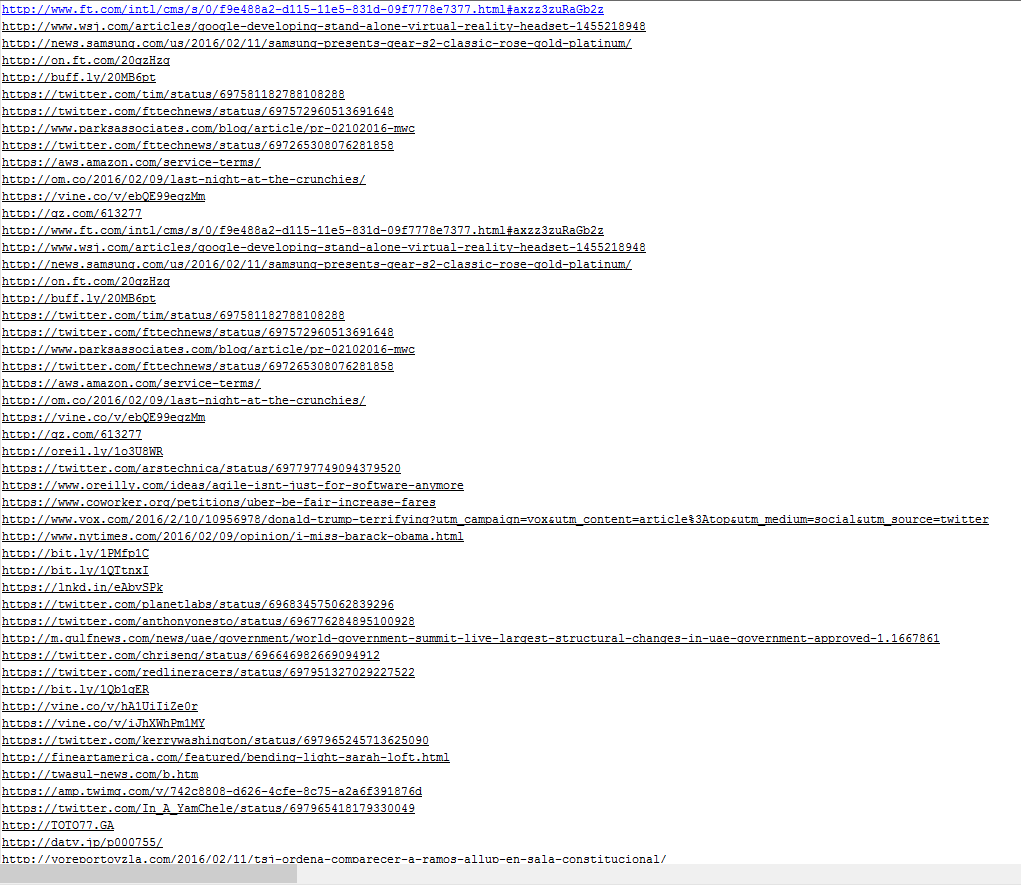
shortened URIs for www.cnn.com (t.co, bit.ly, goo.gl, etc.).

You might want to use the search feature to find URIs, or you can

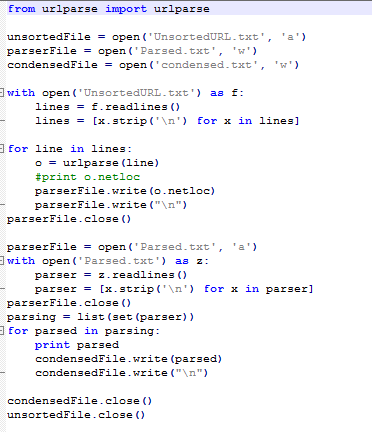
pull them from the feed of someone famous (e.g., Tim O'Reilly).

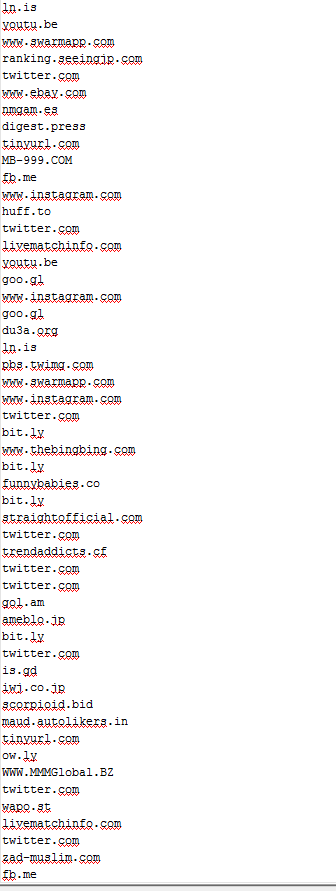


For this I went through and grabbed a specific amount of URI’s from Twitter’s main page. For my sort algorithm I got a total of 7,000 to test with so no duplicates would be found. This all came out in an unsorted list, where each URI had the entire line and there were many occurrences of the same URI. (See next page)

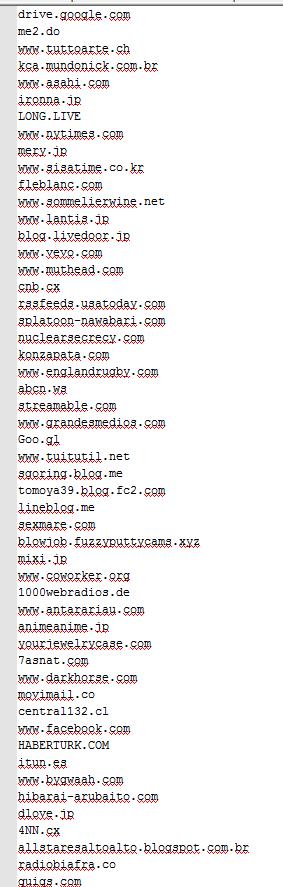


After this step I ran the unsorted list through my condense.py file which took the unsorted list and compared them based off of their net location and then deleted them if they were duplicates:





(Parsed)



At this state, the URI’s were finally condensed entirely. With this I had the thousand URI’s that were needed to get the time map from. They are all stored in the ‘condensed.txt’ file and separated by a new line.

Download the TimeMaps for each of the target URIs. We'll use the mementoweb.org

Aggregator, so for example:

URI-R = http://www.cs.odu.edu/

URI-T = http://mementoweb.org/timemap/link/http://www.cs.odu.edu/

You could use the cs.odu.edu aggregator:

URI-T = http://mementoproxy.cs.odu.edu/aggr/timemap/link/1/http://www.cs.odu.edu/

But be sure to say which aggregator you use -- they are likely to give

different answers.

Create a histogram of URIs vs. number of Mementos (as computed from

the TimeMaps). For example, 100 URIs with 0 Mementos, 300 URIs

with 1 Memento, 400 URIs with 2 Mementos, etc.

See: http://en.wikipedia.org/wiki/Histogram

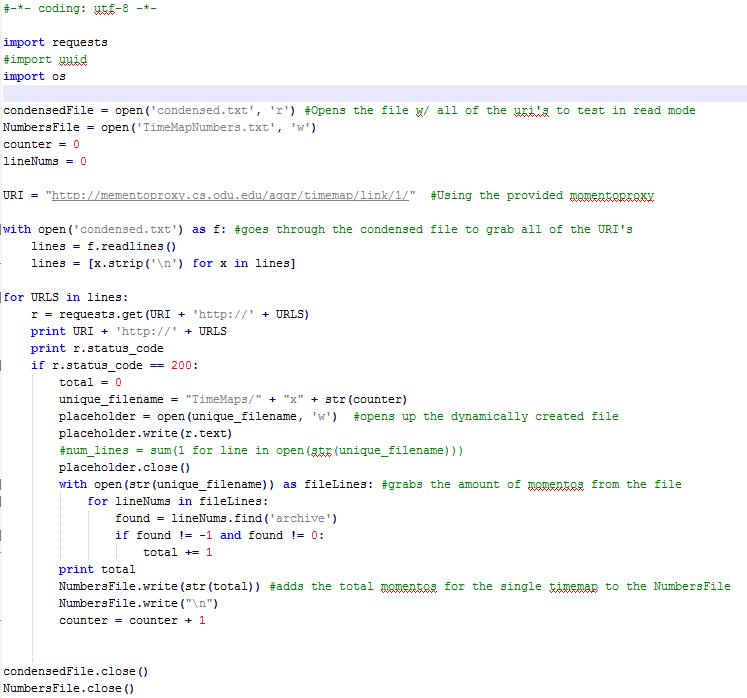
Note that the TimeMaps can span multiple pages. Look for links like:

<http://mementoweb.org/timemap/link/1000/http://www.cnn.com/>;rel="timemap";

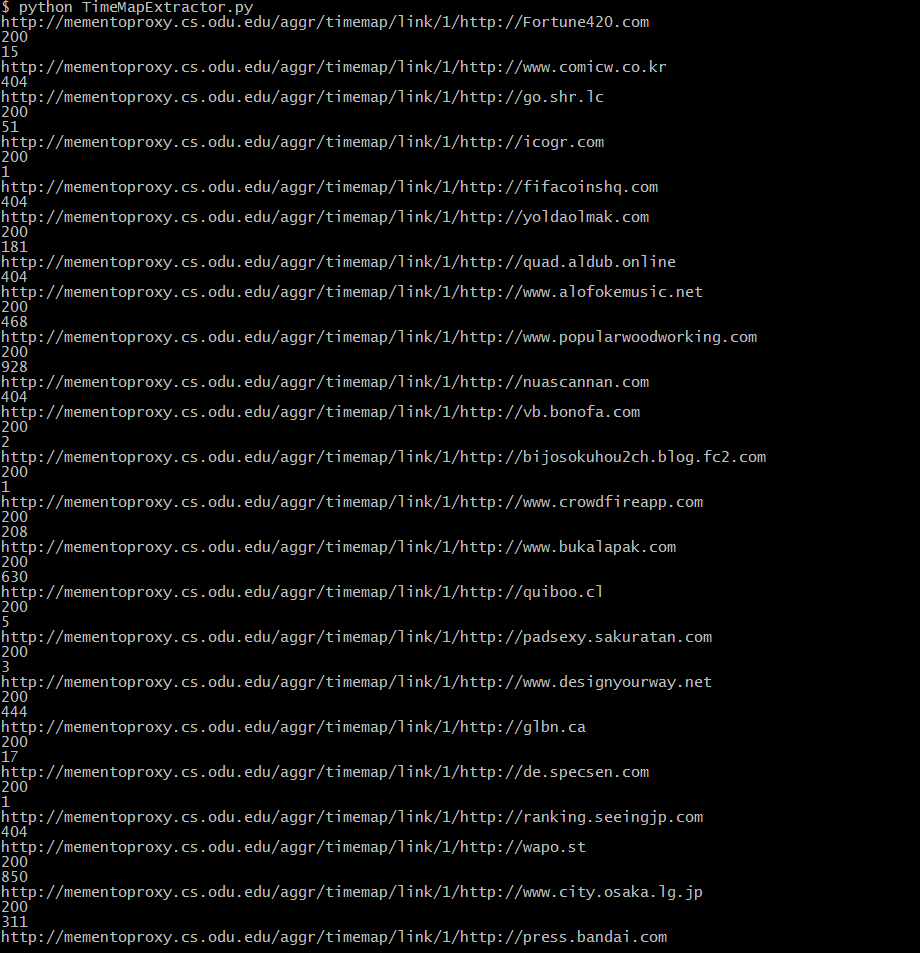
type="application/link-format"; from ="Sun, 08 Jul 2001 21:30:54 GMT"

This indicates another page of the TimeMap is available. There can be

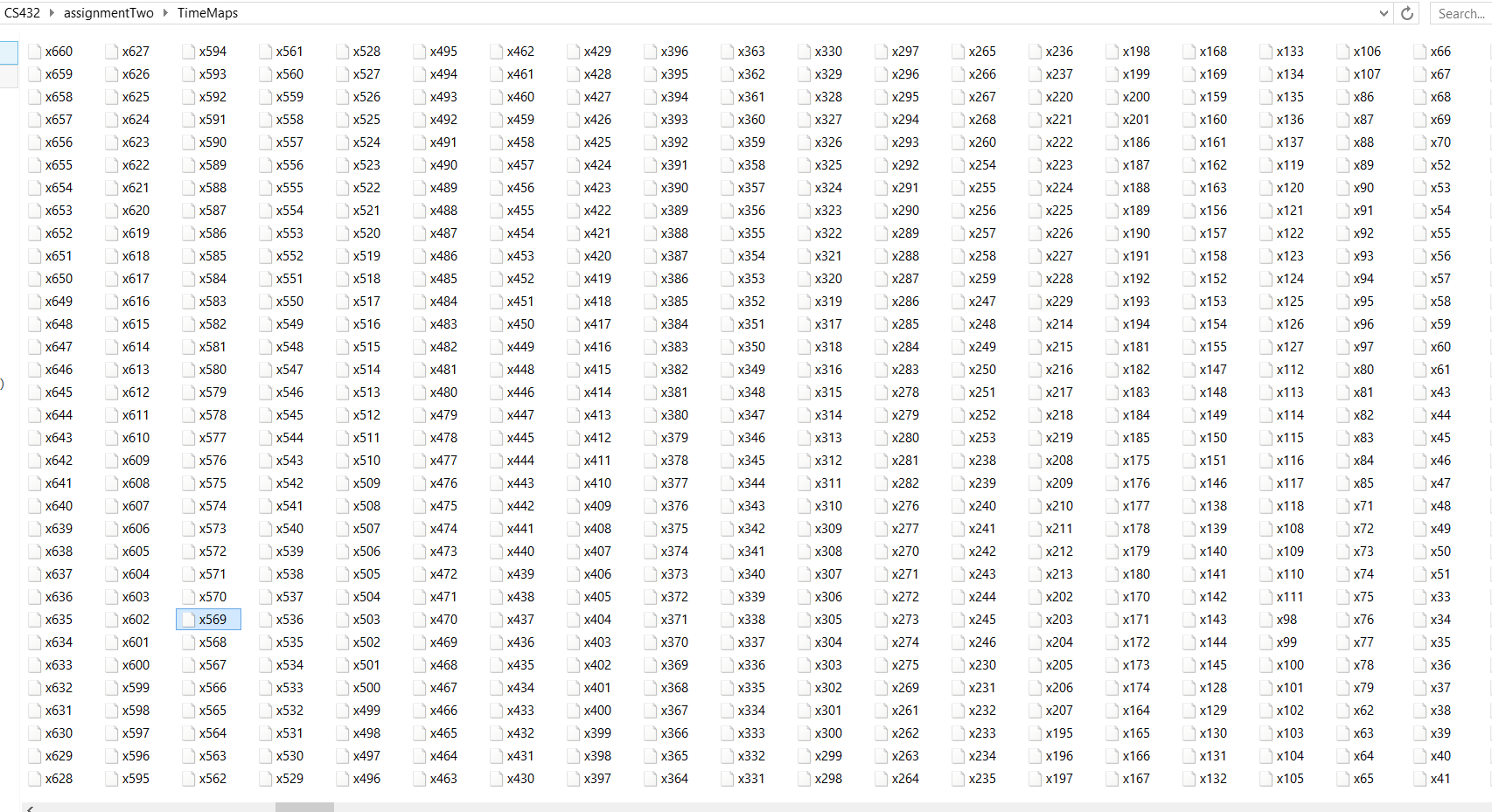
many pages to a TimeMap.

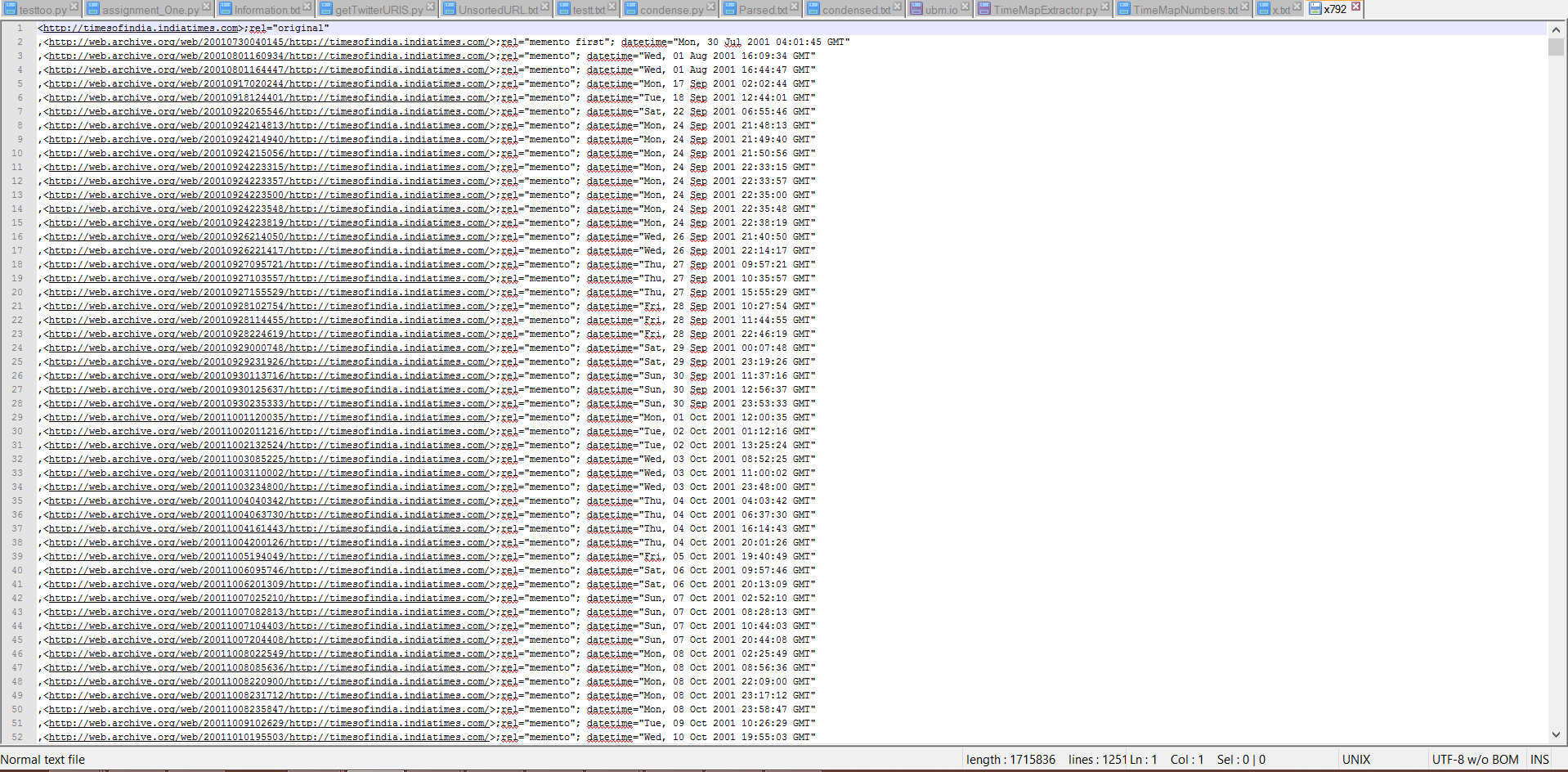


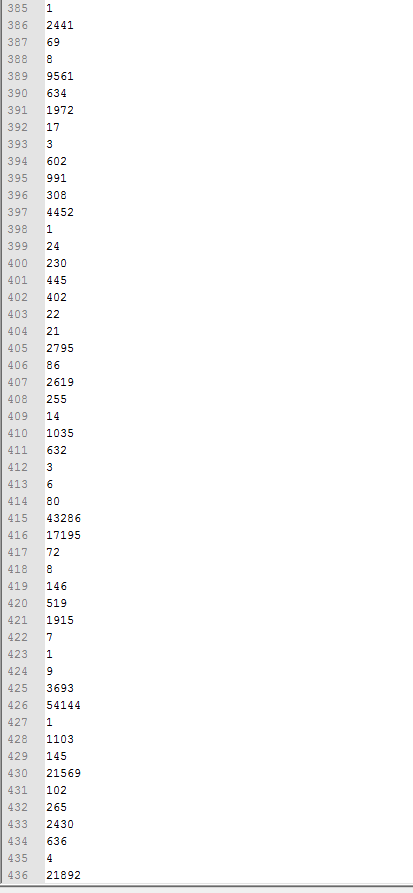
For this part of the project I created a new python program called TimeMapExtractor which grabbed all of the URI’s that I had formatted in the previous problem and ran each of them through the supplied cs.odu.edu momentoproxy URI. The program would make sure that each of the URI’s gave a ‘200’ response code before creating a new file and putting the time map data inside of it. During the compilation step I made sure to add another loop to extract the amount of momentos that each time map had while it was open and stored those numbers in the TimeMapNumbers.txt. When the URI would give a response that was not a 200, I had the program add in a ‘0’ for the amount of momentos that the time map had.



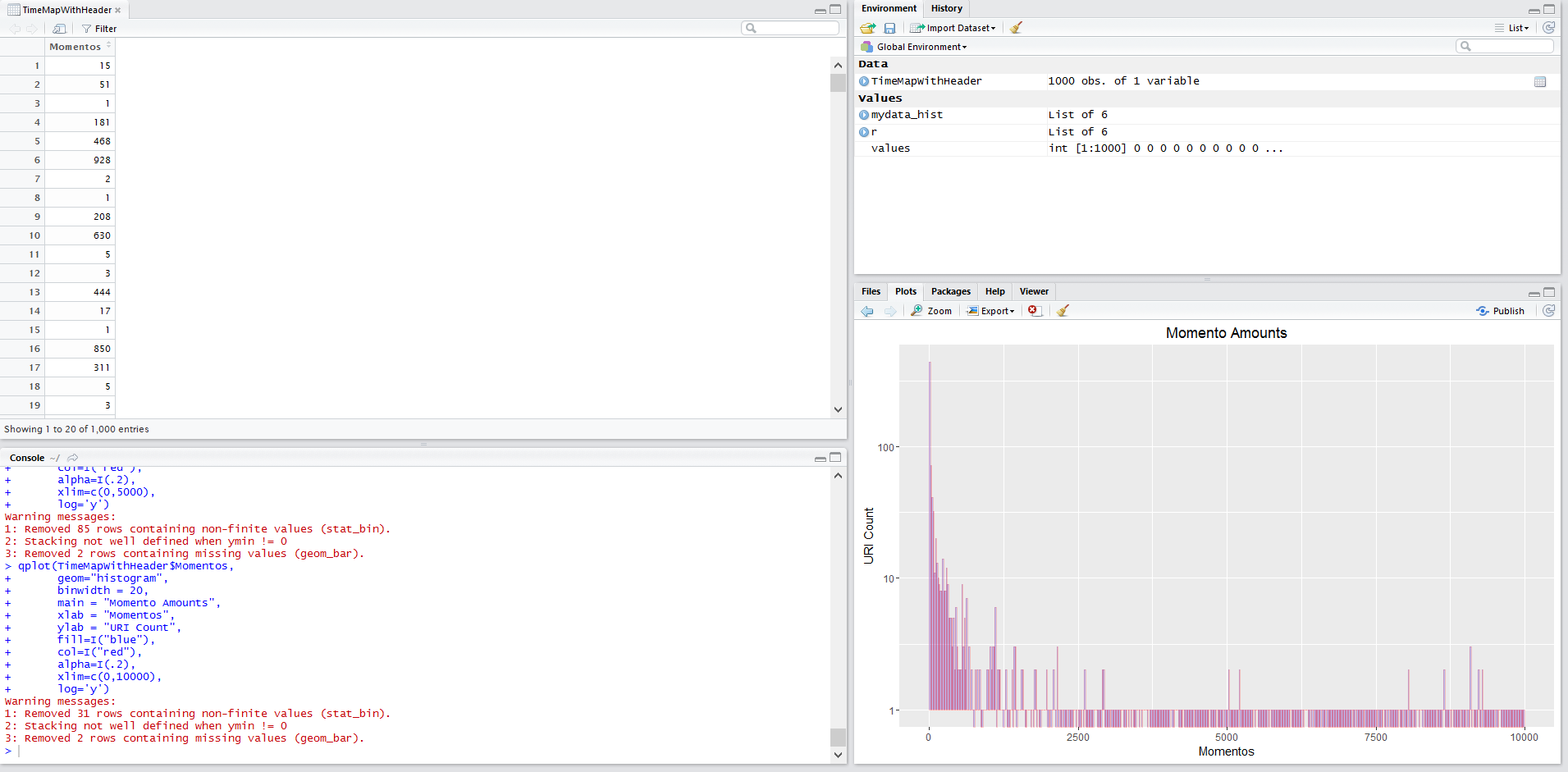
The compilation of TimeMapExtractor.py will show the momentoproxy link along with the response code and the amount of momentos in the file.



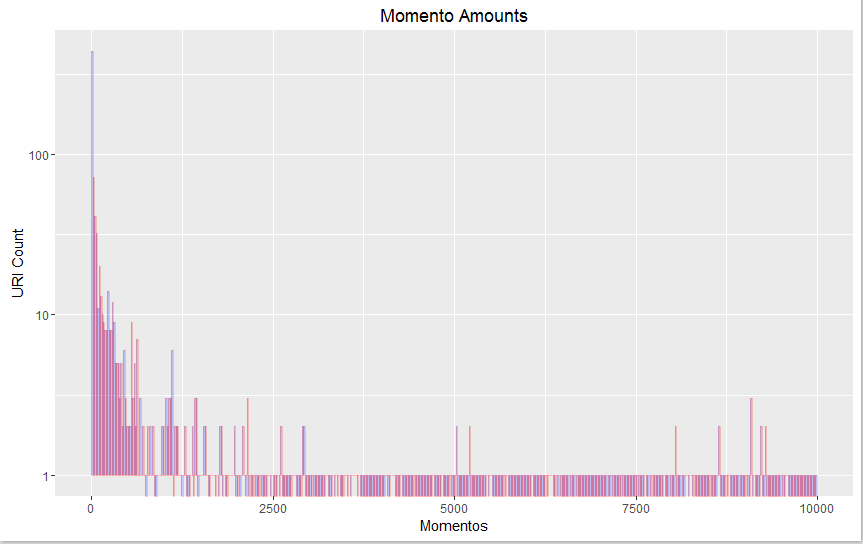
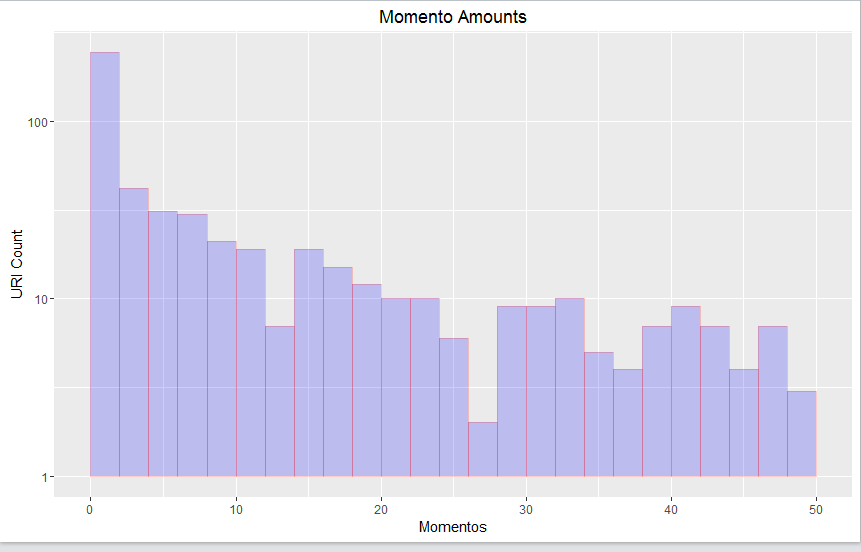
Now, out of the total 1,000 URI’s I started with, a total of 792 ran through the extractor and had a specific amount of momentos. For the other 208 URI’s, a ‘0’ was added into the numbers file (shown below).



Now that I have all of the time map data, I feed the data into R Studio to create the histogram.



After using R Studio to put the data inside of a histogram, I was able to limit the data to two different histograms to model the amount of URI possibilities that I received. This was using the ggplot library built into R Studio.



3. Estimate the age of each of the 1000 URIs using the "Carbon Date" tool:

http://ws-dl.blogspot.com/2014/11/2014-11-14-carbon-dating-web-version-20.html

Note: you'll should download the library and run it locally; don't

try to use the web service.

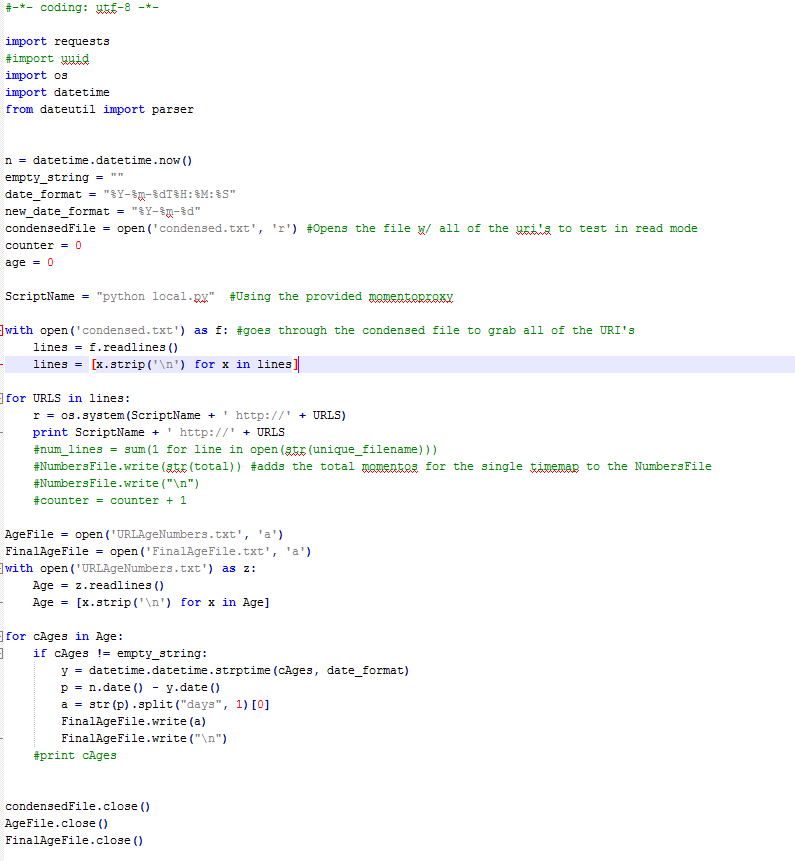
For URIs that have > 0 Mementos and an estimated creation date,

create a graph with age (in days) on one axis and number of mementos

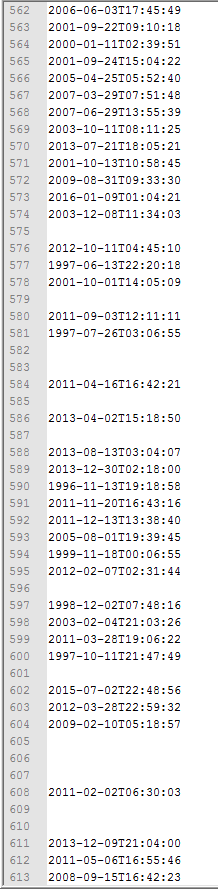
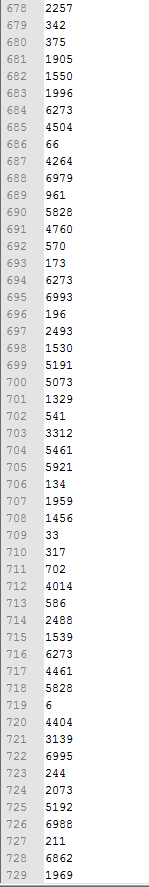
on the other.

Not all URIs will have Mementos, and not all URIs will have an estimated

creation date. State how many fall into either categories.



To get the list of CarbonDates for each of the URI’s that I retrieved in the previous questions I downloaded the CarbonDate tool and ran it locally. I passed in each of the URI’s individually, got the earliest date to which it was created, and parsed the data to just show the amount of days since the creation.



Finally, with the dates received I compared them to the momentos grabbed in question 2 and created a graph in R comparing the two.

