Instructions

Step 1:

Importing all the modules required.

Combining SECFNAME column with https://www.sec.gov/Archives/ and storing it in a new column SECFNAME2.

Step 2:

Extracting data(reports) from links.

I was trying to import all the data in a list name data_source. But the website is showing a error as I am giving 152 responses which is high as per their policies. Read on Google, they were accepting 10 requests per second, hence added sleep time

The output is coming as follows:

```
In [7]: # its showing me a HTML Error

        for i in range(0,len(df_source_links),1):
            response = requests.get(df_source_links[i])
            data_source[i] = response.text
            time.sleep(1)

In [8]: data_source[2]

Out[8]: '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">\n<ht
        ml xmlns="http://www.w3.org/1999/xhtml">\n<head>\n<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />\n<title
        >SEC.gov | Request Rate Threshold Exceeded</title>\n<style>\nhtml {height: 100%}\nbody {height: 100%; margin:0; padding:0;}\n#h
        eader {background-color:#003968; color:#fff; padding:15px 20px 10px 20px;font-family:Arial, Helvetica, sans-serif; font-size:20
        px; border-bottom:solid 5px #000;}\n#footer {background-color:#003968; color:#fff; padding:15px 20px;font-family:Arial, Helveti
        ca, sans-serif; font-size:20px;}\n#content {max-width:650px;margin:60px auto; padding:0 20px 100px 20px; background-image:url(s
        eal_bw.png);background-repeat:no-repeat;background-position:50% 100%;}\nh1 {font-family:Georgia, Times, serif; font-size:20px;}
        \nh2 {text-align:center; font-family:Georgia, Times, serif; font-size:20px; width:100%; border-bottom:solid #999 1px;padding-bo
        ttom:10px; margin-bottom:20px;}\nh3 {font-family:Georgia, Times, serif; font-size:16px; margin:25px 0 0 0;}\np {font-family:Ver
        dana, Geneva, sans-serif;font-size:14px;line-height:1.3;}\n.grey_box {background-color:#eee; padding:5px 40px 20px 40px;margin-
        top:75px;}\n.grey_box p {font-size:12px;line-height:1.5}\n.note {padding: 0 40px; font-style: italic;}\n</style>\n</head>\n\n<b
        ody>\n<div id="header">U.S. Securities and Exchange Commission</div>\n<div id="content">\n<h1>Your Request Originates from an U
        ndeclared Automated Tool</h1>\n<p>To allow for equitable access to all users, SEC reserves the right to limit requests originat
        ing from undeclared automated tools. Your request has been identified as part of a network of automated tools outside of the ac
        ceptable policy and will be managed until action is taken to declare your traffic.</p>\n\n<p>Please declare your traffic by upd
        ating your user agent to include company specific information.</p>\n\n\n<p>For best practices on efficiently downloading inform
        ation from SEC.gov, including the latest EDGAR filings, visit <a href="https://www.sec.gov/developer" target="_blank">sec.gov/d
        eveloper</a>. You can also <a href="https://public.govdelivery.com/accounts/USSEC/subscriber/new?topic_id=USSEC_260" target="_b
        lank">sign up for email updates</a> on the SEC open data program, including best practices that make it more efficient to downl
```

I have tried many ways. Firstly I did it in a single loop like this and also added time sleep to send request by taking a break:

```
In [ ]: # for i in range(0,len(df_source_links),1):
        #     response = requests.get(df_source_links[i])
        #     data_source[i] = response.text
        #     time.sleep(1)
```

Then I tried doing it by 3 loops, but still I got the same data.

```
In [167]: # for i in range(0,8,1):
          #     response = requests.get(df_source_links[i])
          #     data_source[i] = response.text
          #     time.sleep(2)

          # for i in range(8,15,1):
          #     response = requests.get(df_source_links[i])
          #     data_source[i] = response.text
          #     time.sleep(2)

          # for i in range(145,len(df_source_links),1):
          #     response = requests.get(df_source_links[i])
          #     data_source[i] = response.text
          #     time.sleep(2)
```

So, I have extracted 1 data to perform sentiment analysis.

Step 3:

Removing all the punctuations and numbers from the data set.

Tokenizing all the words and sentences in data using NLTK Module and capitalizing it.

Step 4:

Creating a master Stopword list and removing it from tokenized words.

Reading all the dictionary's – Master, Constraining and Uncertainty.

Creating positive and negative dictionary using master dictionary.

Step 5:

Calculating all the scores – Constraining Score, Uncertainty Score, Negative Score, Positive Score, Polarity Score, Subjectivity Score,.

Step 6:

Analysis of readability

Calculating Average Sentence length, Percentage of complex words, Fog Index.

Creating a function to find count of syllable in a word to calculate no. of complex words.

Step 7:

Calculating word count, constraining count, uncertainty count, positive, negative, constraining, uncertainty proportions, and constraining_words_whole_report.

Step 8:

Adding output to XLSX.