# Approach and Though Process for Solution

The code in final.py is a Python script that performs several tasks related to data extraction, cleaning, and analysis, particularly for sentiment and readability.

Throughout the program, I have focused on modularity (with separate functions for different tasks), error handling (e.g., checking for content presence before processing), and efficiency (e.g., using a ThreadPool for concurrent URL processing).

A brief thought process behind each section:  
  
1. **Data Importing**: The script starts by importing necessary libraries and reading an Excel file into a pandas DataFrame. This DataFrame likely contains URLs that will be used for data scraping.  
  
2. **Data Scraping**: It uses BeautifulSoup and requests to scrape content from web pages. The process\_url function fetches the content from each URL, extracts relevant text, and saves it to a text file. It handles cases where the content might be located within different HTML structures.

(2 pages gave 404 error, and were saved as empty files to keep the serial number proper)  
  
3. **Data Cleaning**: The script reads multiple files containing stopwords and combines them into a single list. It then iterates over the text files created during the scraping phase, removes stopwords from the content, and saves the cleaned text to new files.  
  
4. **Sentiment Analysis**: It loads positive and negative words from files into lists and defines functions to calculate positivity, negativity, polarity, and subjectivity scores for the cleaned text.  
  
5. **Readability Analysis**: Several functions are defined to calculate readability metrics such as complex word count, sentence count, average words per sentence, syllable count per word, personal pronouns count, average word length, fog index, and percentage of complex words. These functions are applied to the cleaned text files.  
  
6. **Analysis Execution**: The script executes the sentiment and readability analysis functions on the cleaned text files and prints the results.  
  
7. **Data Aggregation**: It aggregates the results of the analyses into a pandas DataFrame, which is then saved to an Excel file.

8. **Length Verification**: The script checks the lengths of the lists containing analysis results to ensure consistency and identify any discrepancies, which is important for debugging and validation purposes.

(Because initial tests had inconsistencies in length due to some error in code, so did this for validating)  
  
9. **Output**: Finally, the script prints the DataFrame's head to give a preview of the data and saves the DataFrame to an Excel file named readability\_analysis.xlsx

# To run the file

Simply run the final.py file

# Prerequisites to run the code

pip install pandas requests beautifulsoup4 openpyxl

run this line in terminal.