Approach behind the solution:

1. Reading Input Data: The code begins by reading an Excel file (`Input.xlsx`) into a pandas DataFrame (`input_df`). It extracts two columns: `URL` and `URL_ID` from the DataFrame, which represent the web pages to be analyzed and their corresponding identifiers.

2. Extracting Text from URLs: The code uses the `requests` library to fetch the HTML content of each URL and `BeautifulSoup` to parse the HTML. It then extracts the title and the main content (usually in paragraph tags `<p>`) of the web page. The extracted title and content are combined and saved into a text file named after the corresponding `URL_ID`.

3. Text Analysis and Scoring: The code loads the saved text files and analyzes the content using NLP techniques. This analysis involves the following steps:
a. Tokenization: The text is tokenized into words using `nltk` to facilitate word-level analysis.
b. Scoring: Positive and Negative Scores: The code checks how many words in the text match predefined lists of positive and negative words (using `nltk`'s `opinion_lexicon`). Polarity and Subjectivity: Using `TextBlob`, the polarity (how positive or negative the text is) and subjectivity (how subjective or opinionated the text is) scores are calculated.
c. Sentence and Word Counts: Average Sentence Length: Calculated by dividing the total number of words by the number of sentences. Average Word Length: Calculated by dividing the total number of characters by the total number of words.
d. Complex Words and FOG Index: Complex Words: Words with more than two syllables are considered complex. The percentage of complex words is also calculated. FOG Index: A readability test that estimates the years of formal education needed to understand the text. It is calculated using the average sentence length and the percentage of complex words.
e. Personal Pronouns Count: The code counts the occurrence of first-person pronouns (e.g., "I," "we") in the text.
f. Syllable Count per Word: It counts the average number of syllables per word to gauge the complexity of the language used.

4. Saving Results: The results of the text analysis are collected into a list and converted into a pandas DataFrame. This DataFrame is then saved into an Excel file named `Output Data Structure.xlsx`, with columns representing different textual analysis metrics.

Steps to run .py file:

Open the .py file in Google Colab or any other Python IDE and simply run it by clicking the run button.

Dependencies Required:
1. Pandas:
Reading Excel Files: `pd.read_excel()` is used to read data from an Excel file (`Input.xlsx`) into a DataFrame. This allows easy access to and manipulation of the data.
Creating DataFrames: Results from the text analysis are stored in a DataFrame.
Writing Excel Files: `output_df.to_excel()` is used to save the final analysis results into an Excel file (`Output Data Structure.xlsx`).

2. Requests:
Fetching Web Pages: `requests.get(url)` sends a GET request to the specified URL and retrieves the HTML content of the web page.

3. BeautifulSoup (from bs4):
Parsing HTML Content: After fetching the HTML content with `requests`, `BeautifulSoup(response.content, 'html.parser')` is used to parse the HTML and create a soup object.
Extracting Data: The code uses methods like `soup.find('title')` and `soup.find_all('p')` to extract the title and main text content (typically found in `<p>` tags) from the web page.

4. nltk (Natural Language Toolkit):
Word and Sentence Tokenization: `nltk.word_tokenize()` and `nltk.sent_tokenize()` are used to break the text into words and sentences, respectively, which is crucial for subsequent analysis.
Sentiment Lexicons: `nltk.corpus.opinion_lexicon.words('positive-words.txt')` and `nltk.corpus.opinion_lexicon.words('negative-words.txt')` are used to load lists of positive and negative words. These lists are used to calculate the positive and negative scores in the text.
Counting Words and Pronouns: `re.findall()` is used with regular expressions to count occurrences of personal pronouns in the text.

5. TextBlob:
Sentiment Analysis: `TextBlob(text).sentiment` is used to calculate the polarity (a measure of how positive or negative the text is) and subjectivity (a measure of how subjective or opinionated the text is).

6. re (Regular Expressions):
Personal Pronouns Counting: `re.findall(r'\b(?:I|we|my|ours|us)\b', text, re.IGNORECASE)` is used to search for occurrences of personal pronouns in the text, regardless of case.