PROBLEM-2

In [1]:
```
import stanza
stanza.download('en')
nlp = stanza.Pipeline(lang='en', processors='tokenize,pos,constituency' , download_method=None)
# Example sentences
sentences = ["Lucy plays with friends",
             "This movie is careless and unfocused",
             "She buys a gift with gold"]
# Process each sentence and print the constituency parse
for sentence in sentences:
    doc = nlp(sentence)
    print(f"\nSentence: {sentence}")

    # Extract constituent labels if needed
    for sent in doc.sentences:
        # Print the constituency parse tree
        print(sent.constituency.pretty_print())
```

```
c:\users\mano2\appdata\local\programs\python\python37\lib\site-packages\tqdm\auto.py:21: TqdmWarning: IProgress not found. Plea
se update jupyter and ipywidgets. See https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
Downloading https://raw.githubusercontent.com/stanfordnlp/stanza-resources/main/resources_1.6.0.json: 367kB [00:00, 15.1MB/s]
2023-11-20 14:13:49 INFO: Downloading default packages for language: en (English) ...
2023-11-20 14:13:53 INFO: File exists: C:\Users\mano2\stanza_resources\en\default.zip
2023-11-20 14:14:05 INFO: Finished downloading models and saved to C:\Users\mano2\stanza_resources.
2023-11-20 14:14:05 INFO: Loading these models for language: en (English):
=====================================
| Processor    | Package             |
-------------------------------------
| tokenize     | combined            |
| pos          | combined_charlm     |
| constituency | ptb3-revised_charlm |
=====================================

2023-11-20 14:14:05 INFO: Using device: cpu
2023-11-20 14:14:05 INFO: Loading: tokenize
2023-11-20 14:14:05 INFO: Loading: pos
2023-11-20 14:14:06 INFO: Loading: constituency
2023-11-20 14:14:07 INFO: Done loading processors!
```

```
Sentence: Lucy plays with friends
(ROOT
  (S
    (NP (NNP Lucy))
    (VP
      (VBZ plays)
      (PP
        (IN with)
        (NP (NNS friends))))))


Sentence: This movie is careless and unfocused
(ROOT
  (S
    (NP (DT This) (NN movie))
    (VP
      (VBZ is)
      (ADJP (JJ careless) (CC and) (JJ unfocused)))))


Sentence: She buys a gift with gold
(ROOT
  (S
    (NP (PRP She))
    (VP
      (VBZ buys)
      (NP (DT a) (NN gift))
      (PP
        (IN with)
        (NP (NN gold))))))
```

PROBLEM-3

In [2]:
```python
import pandas as pd

# Read the CSV file into a DataFrame
df1 = pd.read_csv('C:/Users/mano2/Downloads/climate change .csv', encoding='cp1252')
df2 = pd.read_csv('C:/Users/mano2/Downloads/Gangs.csv', encoding='cp1252')
df3 = pd.read_csv('C:/Users/mano2/Downloads/Thatcher.csv', encoding='cp1252')
```

```
df1 = df1[df1['Elementary'].notna()]
df2 = df2[df2['Elementary'].notna()]
df3 = df3[df3['Elementary'].notna()]

combined_dataset = pd.concat([df1, df2, df3], ignore_index=True)

# Print the number of rows in the cleaned DataFrame
print("After removing rows with no Elementary text:", len(combined_dataset))
print("\nElementary:\n", combined_dataset['Elementary'][0])
print("\nAdvanced:\n", combined_dataset['Advanced'][0])
```

```
After removing rows with no Elementary text: 35

Elementary:
 Poorer countries will be most affected
by climate change in the next century.
Sea levels will rise, there will be stronger
cyclones, warmer days and nights, more
rainfall, and larger and longer heatwaves,
says a new report.

Advanced:
 Low-income countries will remain on the front
line of human-induced climate change over the
next century, experiencing gradual sea-level
rises, stronger cyclones, warmer days and
nights, more unpredictable rainfall, and larger
and longer heatwaves, according to the most
thorough assessment of the issue yet.
```

PROBLEM-4

In [3]:
```
def count_phrases(tree, phrase_label):
    count = 0
    if tree.label == phrase_label:
        count += 1
    for child in tree.children:
        count += count_phrases(child, phrase_label)
    return count
def parse_texts(texts):
```

```
    return [nlp(text) for text in texts]
def analyze_texts(texts):
    parsed_texts = parse_texts(texts)
    num_sentences = []
    num_pps = []
    num_nps = []
    for doc in parsed_texts:
        sentences = doc.sentences
        num_sentences.append(len(sentences))
        pp_count = sum(count_phrases(sent.constituency, 'PP') for sent in sentences)
        num_pps.append(pp_count)
        np_count = sum(count_phrases(sent.constituency, 'NP') for sent in sentences)
        num_nps.append(np_count)
    avg_sentences = sum(num_sentences) / len(num_sentences)
    avg_pps = sum(num_pps) / len(num_pps)
    avg_nps = sum(num_nps) / len(num_nps)
    return avg_sentences, avg_pps, avg_nps
elementary_texts = combined_dataset['Elementary']#.tolist()
advanced_texts = combined_dataset['Advanced']#.tolist()
elementary_analysis = analyze_texts(elementary_texts)
advanced_analysis = analyze_texts(advanced_texts)
print("Elementary Texts Analysis:", elementary_analysis)
print("Advanced Texts Analysis:", advanced_analysis)
```

```
Elementary Texts Analysis: (3.2285714285714286, 4.857142857142857, 18.17142857142857)
Advanced Texts Analysis: (3.1142857142857143, 6.828571428571428, 21.914285714285715)
```