In [14]:
```
import nltk
```

In [6]:
```
from nltk.tokenize import sent_tokenize, word_tokenize
```

In [3]:
```
example_string = """Muad'Dib learned rapidly because his first training was in how to learn.
And the first lesson of all was the basic trust that he could learn. It's shocking to find how many people do not believe they can
```

In [4]:
```
example_string
```

Out[4]:
```
"Muad'Dib learned rapidly because his first training was in how to learn. \nAnd the first lesson of all was the basic trust that h
e could learn. It's shocking to find how many people do not believe they can learn, and how many more believe learning to be diffi
cult."
```

Sentence Tokenization

In [11]:
```
sent_tokenize(example_string)
```

Out[11]:
```
["Muad'Dib learned rapidly because his first training was in how to learn.",
 'And the first lesson of all was the basic trust that he could learn.',
 "It's shocking to find how many people do not believe they can learn, and how many more believe learning to be difficult."]
```

word Tokenization

In [8]:
```
word_tokenize(example_string)
```

Out[8]:
```
["Muad'Dib",
 'learned',
 'rapidly',
 'because',
 'his',
 'first',
 'training',
 'was',
 'in',
 'how',
 'to',
 'learn',
 '.',
 'And',
 'the',
```

```
 'first',
 'lesson',
 'of',
 'all',
 'was',
 'the',
 'basic',
 'trust',
 'that',
 'he',
 'could',
 'learn',
 '.',
 'It',
 "'s",
 'shocking',
 'to',
 'find',
 'how',
 'many',
 'people',
 'do',
 'not',
 'believe',
 'they',
 'can',
 'learn',
 ',',
 'and',
 'how',
 'many',
 'more',
 'believe',
 'learning',
 'to',
 'be',
 'difficult',
 '.']
```

Filtering Stop Words - Data Cleaning

Stop words are words that you want to ignore, so you filter them out of your text when you’re processing it. Very common words like 'in',

'is', and 'an' are often used as stop words since they don't add a lot of meaning to a text in and of themselves.

In [13]:
```
import nltk
nltk.download("stopwords")
from nltk.corpus import stopwords
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\rakhe\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [15]:
```
worf_quote = "Sir, I protest. I am not a merry man!"
```

In [16]:
```
words_in_quote = word_tokenize(worf_quote)
words_in_quote
```

Out[16]:
```
['Sir', ',', 'I', 'protest', '.', 'I', 'am', 'not', 'a', 'merry', 'man', '!']
```

In [17]:
```
stop_words = set(stopwords.words("english"))
```

In [18]:
```
filtered_list = []
```

In [20]:
```
for words in words_in_quote:
    if words.casefold() not in stop_words:
        filtered_list.append(words)
filtered_list
```

Out[20]:
```
['Sir', ',', 'protest', '.', 'merry', 'man', '!']
```

List Comprehension

In [22]:
```
filtered_list=[words for words in words_in_quote if words.casefold() not in stop_words]
filtered_list
```

Out[22]:
```
['Sir', ',', 'protest', '.', 'merry', 'man', '!']
```

Stemming - is a text processing task in which you reduce words to their root. NLTK has more than one stemmer, but we are using the Porter stemmer.

In [23]:
```
from nltk.stem import PorterStemmer
```

In [24]:
```
stemmer = PorterStemmer()
```

In [25]:
```
string_for_stemming = """The crew of the USS Discovery discovered many discoveries. Discovering is what explorers do."""
```

In [26]:
```
string_for_stemming
```

Out[26]:
```
'The crew of the USS Discovery discovered many discoveries. Discovering is what explorers do.'
```

In [27]:
```
words = word_tokenize(string_for_stemming)
```

In [28]:
```
stemmed_words = [stemmer.stem(word) for word in words]
```

In [29]:
```
stemmed_words
```

Out[29]:
```
['the',
 'crew',
 'of',
 'the',
 'uss',
 'discoveri',
 'discov',
 'mani',
 'discoveri',
 '.',
 'discov',
 'is',
 'what',
 'explor',
 'do',
 '.']
```

In [31]:
```
!pip install porter2stemmer
```

```
Collecting porter2stemmer
```

```
  Downloading porter2stemmer-1.0.tar.gz (14 kB)
Building wheels for collected packages: porter2stemmer
  Building wheel for porter2stemmer (setup.py): started
  Building wheel for porter2stemmer (setup.py): finished with status 'done'
  Created wheel for porter2stemmer: filename=porter2stemmer-1.0-py2.py3-none-any.whl size=6573 sha256=3159cd45751911878ede673600fd
f4af4d59077b727ade53598925a767b1f58e
  Stored in directory: c:\users\rakhe\appdata\local\pip\cache\wheels\86\47\30\c66eb0ceecc9ad2ca83b48c05ef1c1a3a347696f6e7f9f6868
Successfully built porter2stemmer
Installing collected packages: porter2stemmer
Successfully installed porter2stemmer-1.0
```

In [32]:
```
from porter2stemmer import Porter2Stemmer
stemmer = Porter2Stemmer()
print(stemmer.stem('conspicuous'))
```

```
conspicu
```

In [33]:
```
stemmed_words_new = [stemmer.stem(word) for word in words]
```

In [34]:
```
stemmed_words_new
```

Out[34]:
```
['The',
 'crew',
 'of',
 'the',
 'USS',
 'Discoveri',
 'discov',
 'mani',
 'discoveri',
 '.',
 'Discov',
 'is',
 'what',
 'explor',
 'do',
 '.']
```

Part of Speech

In [36]:
```
words
```

Out[36]:
```
['The',
 'crew',
```

```
 'of',
 'the',
 'USS',
 'Discovery',
 'discovered',
 'many',
 'discoveries',
 '.',
 'Discovering',
 'is',
 'what',
 'explorers',
 'do',
 '.']
```

In [45]:
```
lotr_pos_tags = nltk.pos_tag(words)
```

In [46]:
```
lotr_pos_tags
```

Out[46]:
```
[('The', 'DT'),
 ('crew', 'NN'),
 ('of', 'IN'),
 ('the', 'DT'),
 ('USS', 'NNP'),
 ('Discovery', 'NNP'),
 ('discovered', 'VBD'),
 ('many', 'JJ'),
 ('discoveries', 'NNS'),
 ('.', '.'),
 ('Discovering', 'NNP'),
 ('is', 'VBZ'),
 ('what', 'WP'),
 ('explorers', 'NNS'),
 ('do', 'VBP'),
 ('.', '.')]
```

Lemmatization

In [37]:
```
from nltk.stem import WordNetLemmatizer
```

In [42]:
```
lemmatizer = WordNetLemmatizer()
```

In [39]:
```
lemmatized_words = [lemmatizer.lemmatize(word) for word in words]
```

In [40]:
```
lemmatized_words
```

Out[40]:
```
['The',
 'crew',
 'of',
 'the',
 'USS',
 'Discovery',
 'discovered',
 'many',
 'discovery',
 '.',
 'Discovering',
 'is',
 'what',
 'explorer',
 'do',
 '.']
```

Grammer Tree

In [43]:
```
grammar = "NP: {<DT>?<JJ>*<NN>}"
```

In [44]:
```
chunk_parser = nltk.RegexpParser(grammar)
```

In [47]:
```
tree = chunk_parser.parse(lotr_pos_tags)
```

In [ ]:
```
tree.draw()
```

In [ ]:
```

```