In [ ]:
```python
#Import Statements
import nltk
from nltk.book import text1
from nltk.stem import WordNetLemmatizer
import nltk.text
from nltk import word_tokenize
from nltk import sent_tokenize
from nltk import PorterStemmer
```

The code cell above imports all needed functions from nltk

In [ ]:
```python
tokens = text1.tokens[:20]
print(tokens)
```

```
['[', 'Moby', 'Dick', 'by', 'Herman', 'Melville', '1851', ']', 'ETYMOLOGY', '.',
'(', 'Supplied', 'by', 'a', 'Late', 'Consumptive', 'Usher', 'to', 'a', 'Grammar']
```

The code cell above prints out the first 20 tokens of text1. The text object is made out of token objects and tokens are stored as a list.

In [ ]:
```python
print(text1.concordance('sea', 5, 5))
```

```
Displaying 5 of 455 matches:
the sea
Indian Sea
the sea
the sea
the sea
None
```

The code cell above prints out the first 5 instances of sea in text1. Concordance method makes a list of specified words. The cell above finds the first 5 lines that contains the word 'sea'.

NLTK's count method uses python's count method on the text's tokens to find the amount of instances of a given word. They're pretty similar, but with Python's count method you can choose where exactly to look within a list.

In [ ]:
```python
print(text1.count('sea'))
print(text1.tokens.count('sea'))
```

```
433
433
```

Both lines count the instances of sea in text 1. The first line calls the text's count method. The second line calls the token's count method.

In [ ]:
```
raw_text = """With the All Spark gone, we cannot return life to our planet.
And fate has yielded its reward: a new world to call... home.
We live among its people now, hiding in plain sight... but watching over them in se
I have witnessed their capacity for courage, and though we are worlds apart, like u
I am Optimus Prime, and I send this message to any surviving Autobots taking refuge

tokens = word_tokenize(raw_text)
print(tokens[:10])
```

```
['With', 'the', 'All', 'Spark', 'gone', ',', 'we', 'can', 'not', 'return']
```

In [ ]:
```
sentences = sent_tokenize(raw_text)
print(sentences)
```

```
['With the All Spark gone, we cannot return life to our planet.', 'And fate has yie
lded its reward: a new world to call... home.', 'We live among its people now, hidi
ng in plain sight... but watching over them in secret... waiting... protecting.', "
I have witnessed their capacity for courage, and though we are worlds apart, like u
s, there's more to them than meets the eye.", 'I am Optimus Prime, and I send this
message to any surviving Autobots taking refuge among the stars: We are here... we
are waiting.']
```

This code cell separates each sentences with NLTK's sentence tokenizer, then prints them out.

In [ ]:
```
stemText = [PorterStemmer().stem(t) for t in tokens]
print(stemText)
```

```
['with', 'the', 'all', 'spark', 'gone', ',', 'we', 'can', 'not', 'return', 'life',
'to', 'our', 'planet', '.', 'and', 'fate', 'ha', 'yield', 'it', 'reward', ':', 'a',
'new', 'world', 'to', 'call', '...', 'home', '.', 'we', 'live', 'among', 'it', 'peo
pl', 'now', ',', 'hide', 'in', 'plain', 'sight', '...', 'but', 'watch', 'over', 'th
em', 'in', 'secret', '...', 'wait', '...', 'protect', '.', 'i', 'have', 'wit', 'the
ir', 'capac', 'for', 'courag', ',', 'and', 'though', 'we', 'are', 'world', 'apart',
',', 'like', 'us', ',', 'there', "'s", 'more', 'to', 'them', 'than', 'meet', 'the',
'eye', '.', 'i', 'am', 'optimu', 'prime', ',', 'and', 'i', 'send', 'thi', 'messag',
'to', 'ani', 'surviv', 'autobot', 'take', 'refug', 'among', 'the', 'star', ':', 'we
', 'are', 'here', '...', 'we', 'are', 'wait', '.']
```

The code cell above uses the stem function from porter stemmer to remove all the affixes from each token.

In [ ]:
```
lemmatizedText = [WordNetLemmatizer().lemmatize(t) for t in tokens]
print(lemmatizedText)
```

```
['With', 'the', 'All', 'Spark', 'gone', ',', 'we', 'can', 'not', 'return', 'life',
'to', 'our', 'planet', '.', 'And', 'fate', 'ha', 'yielded', 'it', 'reward', ':', 'a
', 'new', 'world', 'to', 'call', '...', 'home', '.', 'We', 'live', 'among', 'it', '
people', 'now', ',', 'hiding', 'in', 'plain', 'sight', '...', 'but', 'watching', 'o
ver', 'them', 'in', 'secret', '...', 'waiting', '...', 'protecting', '.', 'I', 'hav
e', 'witnessed', 'their', 'capacity', 'for', 'courage', ',', 'and', 'though', 'we',
'are', 'world', 'apart', ',', 'like', 'u', ',', 'there', "'s", 'more', 'to', 'them
', 'than', 'meet', 'the', 'eye', '.', 'I', 'am', 'Optimus', 'Prime', ',', 'and', 'I
', 'send', 'this', 'message', 'to', 'any', 'surviving', 'Autobots', 'taking', 'refu
ge', 'among', 'the', 'star', ':', 'We', 'are', 'here', '...', 'we', 'are', 'waiting
', '.']
```

The code cell above uses the lemmatize function from the WordNetLemmatizer to convert each tokens to their root form.

Stem-Lemma

hide-hiding

watch-watching

wait-waiting

protect-protecting

wit-witnessed

The NLTK library is very useful and pretty easy to use, but my only complaint is that downloading and importing all the needed functions can be annoying. The code quality is very good from what I've read, the classes are well organized and there are lots of comments from the developers. I can use NLTK to process text from emails to check whether they're important, use to make a simple chatbot, or to use it for text to speech applications.