Assignment 2

I learned that the tokens() method returns a list of strings (the tokens) and that it's in the context index class and the concordance index class too

In [ ]:

```
from time import process_time_ns
import nltk
from nltk.book import *

# print first 20 tokens from text1

tokens = text1.tokens
for i in range(20): print(tokens[i])
```

```
*** Introductory Examples for the NLTK Book ***
Loading text1, ..., text9 and sent1, ..., sent9
Type the name of the text or sentence to view it.
Type: 'texts()' or 'sents()' to list the materials.
text1: Moby Dick by Herman Melville 1851
text2: Sense and Sensibility by Jane Austen 1811
text3: The Book of Genesis
text4: Inaugural Address Corpus
text5: Chat Corpus
text6: Monty Python and the Holy Grail
text7: Wall Street Journal
text8: Personals Corpus
text9: The Man Who Was Thursday by G . K . Chesterton 1908
[
Moby
Dick
by
Herman
Melville
1851
]
ETYMOLOGY
.
(
Supplied
by
a
Late
Consumptive
Usher
to
a
Grammar
```

In [ ]:

```
text1.concordance('sea', lines=5)
```

```
Displaying 5 of 455 matches:
 shall slay the dragon that is in the sea ." -- ISAIAH " And what thing soever
 S PLUTARCH ' S MORALS . " The Indian Sea breedeth the most and the biggest fis
cely had we proceeded two days on the sea , when about sunrise a great many Wha
many Whales and other monsters of the sea , appeared . Among the former , one w
 waves on all sides , and beating the sea before him into a foam ." -- TOOKE '
```

The nltk count takes the word as the only parameter and counts how many times that appears in the text. The python count method takes the word to count as a parameter and can also take the start and ending indices if called on a string. If called on a list then only the word to count is passed as a parameter. Both methods count the number of times that something is in the text object or list or string.

In [ ]:

```
# the text count method on the text object
print(text1.count('the'))

# the python count on a list
print(tokens.count('the'))
```

```
13721
13721
```

The raw text is the first couple sentences from the Percy Jackson series by Rick Riordan

In [ ]:

```
raw_text = "Look, I didn’t want to be a half-blood. If you’re reading this because you
tokens = nltk.word_tokenize(raw_text)
tokens[0:10]
```

Out[ ]:

```
['Look', ',', 'I', 'didn', '’', 't', 'want', 'to', 'be', 'a']
```

In [ ]:

```
sentences = nltk.sent_tokenize(raw_text)
sentences
```

Out[ ]:

```
['Look, I didn’t want to be a half-blood.',
 'If you’re reading this because you think you might be one, my advice is: close this bo
ok right now.',
 'Believe whatever lie your mom or dad told you about your birth, and try to lead a norm
al life.',
 'Being a half-blood is dangerous.',
 'It’s scary.',
 'Most if the time, it gets you killed in painful, nasty ways']
```

In [ ]:

```
from nltk.stem import PorterStemmer

ps = PorterStemmer()
stemList = [ps.stem(token) for token in nltk.word_tokenize(raw_text)]
stemList
```

Out[ ]:

```
['look',
 ',',
 'i',
 'didn',
 '’',
 't',
 'want',
 'to',
 'be',
 'a',
 'half-blood',
 '.',
 'if',
 'you',
 '’',
 're',
 'read',
 'thi',
 'becaus',
 'you',
 'think',
 'you',
 'might',
 'be',
 'one',
 ',',
 'my',
 'advic',
 'is',
 ':',
 'close',
 'thi',
 'book',
 'right',
 'now',
 '.',
 'believ',
 'whatev',
 'lie',
 'your',
 'mom',
 'or',
 'dad',
 'told',
 'you',
 'about',
 'your',
 'birth',
 ',',
 'and',
 'tri',
 'to',
 'lead',
 'a',
 'normal',
 'life',
 '.',
 'be',
 'a',
 'half-blood',
 'is',
 'danger',
 '.',
 'it',
 '’',
 's',
 'scari',
 '.',
 'most',
 'if',
 'the',
 'time',
 ',',
 'it',
 'get',
 'you',
 'kill',
 'in',
 'pain',
 ',',
 'nasti',
 'way']
```

Differences between the stem and lemma lists: (stem - lemma)

advic - advice

nasti - nasty

pain - painful

scari - scary

whateve - whatever

In [ ]:

```
from nltk.stem import WordNetLemmatizer

nl = WordNetLemmatizer()
lemmaList = [nl.lemmatize(token) for token in nltk.word_tokenize(raw_text)]
lemmaList
```

Out[ ]:

```
['Look',
 ',',
 'I',
 'didn',
 '’',
 't',
 'want',
 'to',
 'be',
 'a',
 'half-blood',
 '.',
 'If',
 'you',
 '’',
 're',
 'reading',
 'this',
 'because',
 'you',
 'think',
 'you',
 'might',
 'be',
 'one',
 ',',
 'my',
 'advice',
 'is',
 ':',
 'close',
 'this',
 'book',
 'right',
 'now',
 '.',
 'Believe',
 'whatever',
 'lie',
 'your',
 'mom',
 'or',
 'dad',
 'told',
 'you',
 'about',
 'your',
 'birth',
 ',',
 'and',
 'try',
 'to',
 'lead',
 'a',
 'normal',
 'life',
 '.',
 'Being',
 'a',
 'half-blood',
 'is',
 'dangerous',
 '.',
 'It',
 '’',
 's',
 'scary',
 '.',
 'Most',
 'if',
 'the',
 'time',
 ',',
 'it',
 'get',
 'you',
 'killed',
 'in',
 'painful',
 ',',
 'nasty',
 'way']
```

NLTK or the natural language toolkit has a lot of funcionality which I find very useful. One of the main features is tokenization which divides the text into smaller units, this can be by words, numbers, punctuation, or sentences. There's other things like stemming and lemming which I find can be very useful when trying to normalize text. The other feature involving stop words is also another great function to help process text. Overall I think that the NLTK library has a lot of good features that make it very functional. I also think that the code quality is really clean and precise when using the NLTK library. Everything is pretty straight forward once you understand what everyhting means and the code is able to do a lot of things in a short length. There are many uses for NLTK in future projects including different ways to process text (ie. tagging parts of speech, filtering stop words, analyzing text)