```
import nltk
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('omw-1.4')
nltk.download('book')
```

```
[nltk_data]    |   Unzipping corpora/ppattach.zip.
[nltk_data]    | Downloading package reuters to /root/nltk_data...
[nltk_data]    | Downloading package senseval to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/senseval.zip.
[nltk_data]    | Downloading package state_union to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/state_union.zip.
[nltk_data]    | Downloading package stopwords to /root/nltk_data...
[nltk_data]    |   Package stopwords is already up-to-date!
[nltk_data]    | Downloading package swadesh to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/swadesh.zip.
[nltk_data]    | Downloading package timit to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/timit.zip.
[nltk_data]    | Downloading package treebank to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/treebank.zip.
[nltk_data]    | Downloading package toolbox to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/toolbox.zip.
[nltk_data]    | Downloading package udhr to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/udhr.zip.
[nltk_data]    | Downloading package udhr2 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/udhr2.zip.
[nltk_data]    | Downloading package unicode_samples to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping corpora/unicode_samples.zip.
[nltk_data]    | Downloading package webtext to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/webtext.zip.
[nltk_data]    | Downloading package wordnet to /root/nltk_data...
[nltk_data]    |   Package wordnet is already up-to-date!
[nltk_data]    | Downloading package wordnet_ic to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/wordnet_ic.zip.
[nltk_data]    | Downloading package words to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/words.zip.
[nltk_data]    | Downloading package maxent_treebank_pos_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping taggers/maxent_treebank_pos_tagger.zip.
[nltk_data]    | Downloading package maxent_ne_chunker to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping chunkers/maxent_ne_chunker.zip.
[nltk_data]    | Downloading package universal_tagset to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping taggers/universal_tagset.zip.
[nltk_data]    | Downloading package punkt to /root/nltk_data...
[nltk_data]    |   Package punkt is already up-to-date!
[nltk_data]    | Downloading package book_grammars to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping grammars/book_grammars.zip.
[nltk_data]    | Downloading package city_database to
[nltk_data]    |     /root/nltk_data...
```

```
[nltk_data]    |   Unzipping corpora/city_database.zip.
[nltk_data]    | Downloading package tagsets to /root/nltk_data...
[nltk_data]    |   Unzipping help/tagsets.zip.
[nltk_data]    | Downloading package panlex_swadesh to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    | Downloading package averaged_perceptron_tagger to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping taggers/averaged_perceptron_tagger.zip.
[nltk_data]    |
[nltk_data]  Done downloading collection book
True
```

Code Below imports all book Text objects

```
from nltk.book import *
```

```
*** 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
```

Extracts the first 20 tokens from text1

```
text1[:20]
```

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

Print a concordance for text1 word 'sea', selecting only 5 lines

```
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 count() method in the API returns the number of occurences of a specified token, whereas the Python count() method returns the number of occurences of any character or string of characters in a string.

```
text1.count("whale")
```

```
906
```

```
text1.count("w")
```

```
0
```

The Raw Text Below will be used to demonstrate the Python count() method plus the methods that follow. It is a segment from Rick Riordan's *Percy Jackson and the Olympians: The Lightning Thief*.

```
raw_text = "Look, I didn't want to be a half-blood. If you're reading this because you think
raw_text.count("Look")
```

```
1
```

```
raw_text.count("L")
```

```
1
```

Tokenize the raw text into word tokens

```
from nltk import word_tokenize
```

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

```
['Look', ',', 'I', 'did', "n't", 'want', 'to', 'be', 'a', 'half-blood']
```

Perform sentence segmentation and display the sentences

```
from nltk import sent_tokenize
```

```
sent_tokens = sent_tokenize(raw_text)
```

```
for i in sent_tokens:
  print(i)
```

```
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 boo
Believe whatever lie your mom or dad told you about your birth, and try to lead a norma
Being a half-blood is dangerous.
It's scary.
Most of the time, it gets you killed in painful, nasty ways.
```

List Comprehension to stem the raw text

```
from nltk.stem.porter import *
```

```
stemmer = PorterStemmer()
stemmed = [stemmer.stem(t) for t in tokens]
print(stemmed)
```

```
ou', 'think', 'you', 'might', 'be', 'one', ',', 'my', 'advic', 'is', ':', 'close', 'thi'
```

List Comprehension to lemmatize the raw text

```
from nltk.stem import WordNetLemmatizer
```

```
wnl = WordNetLemmatizer()
lemmatized = [wnl.lemmatize(t) for t in tokens]
print(lemmatized)
```

```
ng', 'this', 'because', 'you', 'think', 'you', 'might', 'be', 'one', ',', 'my', 'advice',
```

Stem vs Lemmas

1. look - Look
2. read - reading
3. thi - this
4. advic - advice
5. thi - this

The NLTK library has very intuitive functions and I've learned that it can be extremely useful in text processing and analyzing. The code quality of the functions is high. Perhaps there is some code I could use to optimize some of these functions that I am not yet aware of because I am an NLTK beginner, but the token method turned contraction words into two separate tokens before and after the apostrophe. This is a detail of the code that I believe can be improved. I can use NLTk in future projects, such as those having to do with analyzing text sentiments. For example:

- Learning the key words in Social Media Posts
- Reviewing text books for students to learn most used key words
- Creating word clouds to analyze the tone of Books and novels
- Associating Movie Titles with famous opening lines