```
import nltk
```

2.

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

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...
True
```

```
nltk.download('book')
```

```
[nltk_data] Downloading collection 'book'
[nltk_data]    |
[nltk_data]    | Downloading package abc to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/abc.zip.
[nltk_data]    | Downloading package brown to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/brown.zip.
[nltk_data]    | Downloading package chat80 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/chat80.zip.
[nltk_data]    | Downloading package cmudict to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/cmudict.zip.
[nltk_data]    | Downloading package conll2000 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/conll2000.zip.
[nltk_data]    | Downloading package conll2002 to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/conll2002.zip.
[nltk_data]    | Downloading package dependency_treebank to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping corpora/dependency_treebank.zip.
[nltk_data]    | Downloading package genesis to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/genesis.zip.
[nltk_data]    | Downloading package gutenberg to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/gutenberg.zip.
[nltk_data]    | Downloading package ieer to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/ieer.zip.
[nltk_data]    | Downloading package inaugural to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/inaugural.zip.
[nltk_data]    | Downloading package movie_reviews to
[nltk_data]    |     /root/nltk_data...
[nltk_data]    |   Unzipping corpora/movie_reviews.zip.
[nltk_data]    | Downloading package nps_chat to /root/nltk_data...
[nltk_data]    |   Unzipping corpora/nps_chat.zip.
[nltk_data]    | Downloading package names to /root/nltk_data...
```

```
[nltk_data]    |   Unzipping corpora/names.zip.
[nltk_data]    | Downloading package ppattach to /root/nltk_data...
[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.
```

```
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
```

3. -tokens() becomes a list

-Text object has built in methods including tokens()

```
tokens1 = text1.tokens[:20]
tokens1
```

```
['[',
 'Moby',
```

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

4. prints 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 '
```

5. The count() method in the API and Python's are similar methods that count the number of times the words/object appears in the text/list

```
print(text1.count("sea"))
list1 = ["sea", "test", "word", "sea", "listtest","counting", "blue"]
print(list1.count("sea"))
```

```
433
2
```

6. word_tokenize converts text to tokens, which can then be printed as a list.

Citation for raw text: Orwell, George, and Kamoun Josée. 1984. Gallimard, 2020.

```
raw_text = "It was a bright cold day in April, and the clocks were striking thirteen. Winston
```

```
from nltk import word_tokenize
```

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

```
['It', 'was', 'a', 'bright', 'cold', 'day', 'in', 'April', ',', 'and']
```

7. imports the method and tokenizes the text into sentences which then can be printed

```
from nltk import sent_tokenize
sentences = sent_tokenize(raw_text)
for sentence in sentences:
  print(sentence)
```

```
It was a bright cold day in April, and the clocks were striking thirteen.
Winston Smith, his chin nuzzled into his breast in an effort to escape the vile wind, sl
The hallway smelt of boiled cabbage and old rag mats.
At one end of it a coloured poster, too large for indoor display, had been tacked to the
It depicted simply an enormous face, more than a metre wide: the face of a man of about
Winston made for the stairs.
It was no use trying the lift.
Even at the best of times it was seldom working, and at present the electric current was
It was part of the economy drive in preparation for Hate Week.
The flat was seven flights up, and Winston, who was thirty-nine and had a varicose ulcer
On each landing, opposite the lift-shaft, the poster with the enormous face gazed from t
It was one of those pictures which are so contrived that the eyes follow you about when
BIG BROTHER IS WATCHING YOU, the caption beneath it ran.
```

8. used the PorterStemmed() method to create a list stemmed for each token, which is then printed

```
from nltk.stem.porter import *
stemmer = PorterStemmer()
stemmed = [stemmer.stem(t) for t in tokens]
print('stemmed tokens:\n', stemmed)
```

```
stemmed tokens:
 ['it', 'wa', 'a', 'bright', 'cold', 'day', 'in', 'april', ',', 'and', 'the', 'clock',
```

9. Code works similarly as using the PorterStemmer. Differences:

-stem does not keep capitalization-lemma does -stem shortens words-lemma keeps the whole thing -stem will sometimes convert a word into the base word-lemma keeps them unchanged -stem replaced 'y's with i-lemma kept it -some stem words are hard to differentiate like 'as' gets changed to 'a'-lemma again keeps words unchanged

```
from nltk.stem import WordNetLemmatizer
wnl = WordNetLemmatizer()
lemmatized = [wnl.lemmatize(t) for t in tokens]
print('lemmatized tokens:\n', lemmatized)
```

```
lemmatized tokens:
 ['It', 'wa', 'a', 'bright', 'cold', 'day', 'in', 'April', ',', 'and', 'the', 'clock',
```

The functionality seems relativility simple and straight forward, would like to see some more features but to be fair I haven't explored it enough. The code seems fine, its well readable and writable. NLTK can be used to tokenize and pull apart meanings of words and sentences a lot easier.