In [1]:

```python
import nltk
from nltk.book import *
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.stem.porter import *
from nltk.stem import WordNetLemmatizer
```

```
*** 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
```

Things learned about Text objects

1. Text objects are typically initialized from a given document or corpus.
2. The tokens attribute returns a list

Print first 20 tokens of text1

In [2]:

```python
text1.tokens[:20]
```

Out[2]:

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

Shows occurence of the word 'sea' in first 5 lines

In [3]:

```python
text1.concordance(word='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 '
```

Comparing count() methods

count() counts the number of times a word appears in the text. Both count() methods are the same in that they both take in one value to return the number of times it is found in the text. The built-in python count() takes in two optional parameters, start and end, that allows the user specify where to start and stop the search.

In [4]:

```python
print(text1.count('life'))
print(text2.count('death'))
```

```
162
13
```

raw_text taken from *Moby Dick*

Tokenize text into words

In [5]:

```python
raw_text = "Call me Ishmael. Some years ago — never mind how long precisely — having little or no money in my p
word_tokenize(raw_text)[:10]
```

Out[5]:

```
['Call', 'me', 'Ishmael', '.', 'Some', 'years', 'ago', '—', 'never', 'mind']
```

Tokenize text into sentences

In [6]:

```python
sent_tokenize(raw_text)[:10]
```

Out[6]:

```
['Call me Ishmael.',
 'Some years ago — never mind how long precisely — having little or no money in my purse, and nothing particula
r to interest me on shore, I thought I would sail about a little and see the watery part of the world.',
 'It is a way I have of driving off the spleen, and regulating the circulation.']
```

In [7]:

```python
stemmer = PorterStemmer()
tokens = word_tokenize(raw_text)
stemmed_tokens = [stemmer.stem(tok) for tok in tokens]
print(stemmed_tokens)
```

```
['call', 'me', 'ishmael', '.', 'some', 'year', 'ago', '—', 'never', 'mind', 'how', 'long', 'precis', '—', 'hav
e', 'littl', 'or', 'no', 'money', 'in', 'my', 'purs', ',', 'and', 'noth', 'particular', 'to', 'interest', 'me',
'on', 'shore', ',', 'i', 'thought', 'i', 'would', 'sail', 'about', 'a', 'littl', 'and', 'see', 'the', 'wateri',
'part', 'of', 'the', 'world', '.', 'it', 'is', 'a', 'way', 'i', 'have', 'of', 'drive', 'off', 'the', 'spleen',
',', 'and', 'regul', 'the', 'circul', '.']
```

Comparing stemmed vs lemmatized text

1. Voldemort vs voldemort
2. creat vs created
3. hi vs his
4. enemi vs enemy
5. as vs a

In [8]:

```python
lemmatizer = WordNetLemmatizer()
tokens = word_tokenize(raw_text)
lem_tokens = [lemmatizer.lemmatize(tok) for tok in tokens]
print(lem_tokens)
```

```
['Call', 'me', 'Ishmael', '.', 'Some', 'year', 'ago', '—', 'never', 'mind', 'how', 'long', 'precisely', '—', 'h
aving', 'little', 'or', 'no', 'money', 'in', 'my', 'purse', ',', 'and', 'nothing', 'particular', 'to', 'interes
t', 'me', 'on', 'shore', ',', 'I', 'thought', 'I', 'would', 'sail', 'about', 'a', 'little', 'and', 'see', 'th
e', 'watery', 'part', 'of', 'the', 'world', '.', 'It', 'is', 'a', 'way', 'I', 'have', 'of', 'driving', 'off',
'the', 'spleen', ',', 'and', 'regulating', 'the', 'circulation', '.']
```

a. Opinion of functionality of NLTK

NLTK is intuitive and easy to use. The documentation is well written and there are many methods that provide a wide variety of functions that allow for effective analysis of text of any size.

b. Opinion on code quality of NLTK library

Based on the nltk.Text API, the code is written well and is easy and efficient to use.

c. Future use of NLTK

I hope to use nltk libraries in sentiment analysis projects for text preprocessing and analysis.