Q1 & Q2

Made the Jupyter notebook file and have NLTK installed

Q3

Printing first 20 tokens from the text1 NLTK Text Object with tokens()

1. Tokens() can use context or concordance index and returns that document.
2. Text object has two separate classes for ContextIndex(uses bidirectional index between words or by searching within the fixed area of a word) and ConcordanceIndex(uses offset location to find words)

In [175]:

```
import nltk
from nltk.book import *
tok = text1[:20]
print(tok)
```

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

Q4

concordance() with the string 'sea'

In [176]:

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

```
Displaying 5 of 455 matches:
 shall slay the dragon that is in the sea ." -- ISAIAH " And what thing soeve
r
 S PLUTARCH ' S MORALS . " The Indian Sea breedeth the most and the biggest f
is
cely had we proceeded two days on the sea , when about sunrise a great many W
ha
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
'
```

Q5

count() in NLTK vs built in library NLTK count takes in only NLTK text objects Built in count takes in lists to count

In [177]:

```
print('Count in NLTK - ' + str(text1.count('sea')))
ls = ['hello', 'this', 'is','Sharar', 1,2,3,4,5,6,7,8,1,2,3,4,5,6,7,8]
```

```
print(f'Count in built in python functionality - {ls.count(1)}')
```

```
Count in NLTK - 433
Count in built in python functionality - 2
```

Q6

word_tokenize() tokeninzes to words given some text

In [178]:

```
from nltk import word_tokenize

#source of data lore from Titanfall game
https://titanfall.fandom.com/wiki/Titanfall_Universe
lore = 'The Titanfall Universe is the fictional universe in which the
Titanfall series takes place. In the distant future, humanity has discovered
the means of faster-than-light travel and has proceeded to colonize deep
space. The first waves of colonists colonized several star systems close to
Sol, these colonies becoming known as the Core Systems. However, exploration
did not stop there, and soon, a dense region of habitable world was
discovered; ripe with resources and valuable trade routes, this new Frontier
would soon become home to many generations of humanity. However, the arrival
of the Interstellar Manufacturing Corporation would tip the balance of power
on the Frontier, leading to the formation of the Frontier Militia and the
beginning of the Frontier War - the conflict in which the Titanfall series
takes place.'
tokens = word_tokenize(lore)
print(tokens[:10])
```

```
['The', 'Titanfall', 'Universe', 'is', 'the', 'fictional', 'universe', 'in',
'which', 'the']
```

Q7

sent_tokenize() tokeninzes to sentences given some text

In [179]:

```
from nltk import sent_tokenize
tokens = sent_tokenize(lore)
tokens
```

Out[179]:

```
['The Titanfall Universe is the fictional universe in which the Titanfall ser
ies takes place.',
 'In the distant future, humanity has discovered the means of faster-than-lig
ht travel and has proceeded to colonize deep space.',
 'The first waves of colonists colonized several star systems close to Sol, t
hese colonies becoming known as the Core Systems.',
 'However, exploration did not stop there, and soon, a dense region of habita
ble world was discovered; ripe with resources and valuable trade routes, this
new Frontier would soon become home to many generations of humanity.',
 'However, the arrival of the Interstellar Manufacturing Corporation would ti
p the balance of power on the Frontier, leading to the formation of the Front
```

```
ier Militia and the beginning of the Frontier War - the conflict in which the
Titanfall series takes place.']
```

Q8

NLTK's PorterStemmer() can be used to reduce words to its base form But in many cases it may give incorrect info as it disregards the context

In [180]:

```
from nltk.stem.porter import *
stemmer = PorterStemmer()
words = [ 'waits', 'runs', 'ability', 'flies', 'cries', 'initializes',
'loftily','violently',
'rightfully','seldom','anxiously','frequents','smoothly','safety']
core = [stemmer.stem(stemmed_word) for stemmed_word in words]
print(core)
```

```
['wait', 'run', 'abil', 'fli', 'cri', 'initi', 'loftili', 'violent', 'right',
'seldom', 'anxious', 'frequent', 'smoothli', 'safeti']
```

Q9

NLTK's WordNetLemmatizer() can be used to reduce words to its base form This accounts for the context and gives more accurate base form

|---|---|---|

| **Word** | **Stemmed** | **Lemmatized** |

| flies | fli | fly |

| cries | cri | cry |

| safety | safeti | safety |

| runs | run | run |

| frequents | frequent | frequents |

| smoothly | smoothli | smoothly |

In [181]:

```
from nltk.stem.wordnet import *
lemmatizer = WordNetLemmatizer()
core = [lemmatizer.lemmatize(lemmatized_word) for lemmatized_word in words]
print(core)
```

```
['wait', 'run', 'ability', 'fly', 'cry', 'initializes', 'loftily', 'violently
', 'rightfully', 'seldom', 'anxiously', 'frequents', 'smoothly', 'safety']
```

Q10

The NLTK Library is a great libray for learning and using in NLP. The library has many functionalities that can help us extract words. Then we can use those words anyway we want to. Give proper responses, extract data, train AI etc.

The NLTK library is well written. The code can be understood easily and the documentation on it is fairly simple. It would be better if we had some example code to refer to.

This library opens up a lot of possibilities for use in future projects. A few ways of the using it are,

1. Use NLTK to extract data from a projects ReadMe.md and use the top 10 keywords to add tags for the said projects github pages.
2. Use the extracted data to train AI for different languages.
3. Use it to do sentiment analysis on user input in future projects