```
import nltk
from nltk import word_tokenize
from nltk import sent_tokenize
nltk.download('gutenberg')
nltk.download('stopwords')
nltk.download('wordnet')
nltk.download('punkt')
nltk.download('omw-1.4')
nltk.download('genesis')
nltk.download('inaugural')
nltk.download('nps_chat')
nltk.download('webtext')
nltk.download('treebank')
```

```
[nltk_data] Downloading package gutenberg to /root/nltk_data...
[nltk_data]   Package gutenberg is already up-to-date!
[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...
[nltk_data]   Package omw-1.4 is already up-to-date!
[nltk_data] Downloading package genesis to /root/nltk_data...
[nltk_data]   Package genesis is already up-to-date!
[nltk_data] Downloading package inaugural to /root/nltk_data...
[nltk_data]   Package inaugural is already up-to-date!
[nltk_data] Downloading package nps_chat to /root/nltk_data...
[nltk_data]   Package nps_chat is already up-to-date!
[nltk_data] Downloading package webtext to /root/nltk_data...
[nltk_data]   Package webtext is already up-to-date!
[nltk_data] Downloading package treebank to /root/nltk_data...
[nltk_data]   Package treebank is already up-to-date!
True
```

Import the text tokens to be called on in the following code segnments

```
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
```

Print the first 20 tokens from text1 aka Moby Dick by Herman Melville 1851. An interesting thing I learned about tokenization is how the nltk software is sophisticated enough to split a body of texts into tokens on its own. It's also interesting how we are able to filter through these tokens so easily.

```
print(text1[:20])
```

```
['[', 'Moby', 'Dick', 'by', 'Herman', 'Melville', '1851', ']', 'ETYMOLOGY', '.',
```

Look through the Moby Dick text and find the first 5 sentences that contains the word sea, and print it

```
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 '
```

Use pythons internal count method to count characters then use NLTK count() to see how many times the phrase moby comes up. Count() in nltk gives you the number of tokens including characters such as '(' or '.' Instead of just giving you the length like python's count method.

```
x = len(text1)
y =text1.count("Moby")

print(x)
print(y)
```

```
260819
84
```

Using NLTK's word tokenizer totokenize some raw text into a variable namd tokens. Print the first 10 tokens.

```
raw_text = "Natural language processing (NLP) is a subfield of linguistics, computer s
        "The goal is a computer capable of understanding the contents of documents, in
        "The technology can then accurately extract information and insights containec
        "Natural language processing has its roots in the 1950s. Already in 1950, Alar
```

```
tokens = nltk.word_tokenize(raw_text)

print(tokens[:10])

    ['Natural', 'language', 'processing', '(', 'NLP', ')', 'is', 'a', 'subfield', 'o
```

Using the same raw text to make sentence tokens using NLTK's sentence tokenizer sent_tokenize(). Display the sentences.

```
tokens2 = nltk.sent_tokenize(raw_text)

print(tokens2)

    ['Natural language processing (NLP) is a subfield of linguistics, computer scien
```

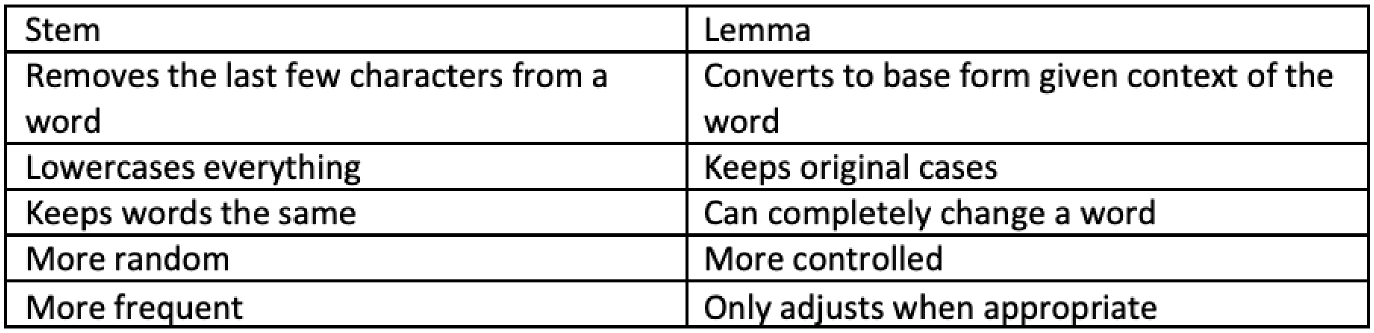
Using NLTK's PorterStemmer() to write a list comprehension to stem the text. Display the list.

```
from nltk.stem import *
from nltk.stem.porter import *
stemmer = PorterStemmer()

tokens = nltk.word_tokenize(raw_text)
stemTest = [stemmer.stem(w) for w in tokens]
print(stemTest)

    ['natur', 'languag', 'process', '(', 'nlp', ')', 'is', 'a', 'subfield', 'of', 'l
```

Using NLTK's WordNetLemmatizer, write a list comprehension to lemmatize the text. Display the list.



Stem	Lemma
Removes the last few characters from a word	Converts to base form given context of the word
Lowercases everything	Keeps original cases
Keeps words the same	Can completely change a word
More random	More controlled
More frequent	Only adjusts when appropriate

```
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()

tokens = nltk.word_tokenize(raw_text)
lemmaText = [lemmatizer.lemmatize(w) for w in tokens]
print(lemmaText)
```

```
['Natural', 'language', 'processing', '(', 'NLP', ')', 'is', 'a', 'subfield', 'o
```

NLTK streamlines the ability to gather and analyze texts in python. This makes designing algorithms much easier. One drawback is that nltk does much the work for you. Because of this, you get much less control on exactly how you want to gather and manipulate text. For a large body of data, this is not as imporant. But if you want to be precise with a small amount of text, this can be very difficult. I can see myself using nltk to gather some key words and determine the genre of the text based on how many occurences of these key words.