In [1]:
```
import nltk
```

In [2]:
```
nltk.download()
```

```
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml (https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml)
```

Out[2]:
```
True
```

In [3]:
```
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
```

In [4]:
```
from nltk.corpus import brown
```

In [5]:
```
brown.categories()
```

Out[5]:
```
['adventure',
 'belles_lettres',
 'editorial',
 'fiction',
 'government',
 'hobbies',
 'humor',
 'learned',
 'lore',
 'mystery',
 'news',
 'religion',
 'reviews',
 'romance',
 'science_fiction']
```

In [6]:
```
brown.words(categories = 'adventure')[:50]
```

Out[6]:
```
['Dan',
 'Morgan',
 'told',
 'himself',
 'he',
 'would',
 'forget',
 'Ann',
 'Turner',
 '.',
 'He',
 'was',
 'well',
 'rid',
 'of',
 'her',
 '.',
 'He',
 'certainly',
 "didn't",
 'want',
 'a',
 'wife',
 'who',
 'was',
 'fickle',
 'as',
 'Ann',
 '.',
 'If',
 'he',
 'had',
 'married',
 'her',
 ',',
 "he'd",
 'have',
 'been',
 'asking',
 'for',
 'trouble',
 '.',
 'But',
 'all',
 'of',
 'this',
 'was',
 'rationalization',
 '.',
 'Sometimes']
```

In [7]:
```
from nltk.corpus import inaugural
```

In [8]:
```
inaugural.fileids()
```

Out[8]:
```
['1789-Washington.txt',
 '1793-Washington.txt',
 '1797-Adams.txt',
 '1801-Jefferson.txt',
 '1805-Jefferson.txt',
 '1809-Madison.txt',
 '1813-Madison.txt',
 '1817-Monroe.txt',
 '1821-Monroe.txt',
 '1825-Adams.txt',
 '1829-Jackson.txt',
 '1833-Jackson.txt',
 '1837-VanBuren.txt',
 '1841-Harrison.txt',
 '1845-Polk.txt',
 '1849-Taylor.txt',
 '1853-Pierce.txt',
 '1857-Buchanan.txt',
 '1861-Lincoln.txt',
 '1865-Lincoln.txt',
 '1869-Grant.txt',
 '1873-Grant.txt',
 '1877-Hayes.txt',
 '1881-Garfield.txt',
 '1885-Cleveland.txt',
 '1889-Harrison.txt',
 '1893-Cleveland.txt',
 '1897-McKinley.txt',
 '1901-McKinley.txt',
 '1905-Roosevelt.txt',
 '1909-Taft.txt',
 '1913-Wilson.txt',
 '1917-Wilson.txt',
 '1921-Harding.txt',
 '1925-Coolidge.txt',
 '1929-Hoover.txt',
 '1933-Roosevelt.txt',
 '1937-Roosevelt.txt',
 '1941-Roosevelt.txt',
 '1945-Roosevelt.txt',
 '1949-Truman.txt',
 '1953-Eisenhower.txt',
 '1957-Eisenhower.txt',
 '1961-Kennedy.txt',
 '1965-Johnson.txt',
 '1969-Nixon.txt',
 '1973-Nixon.txt',
 '1977-Carter.txt',
 '1981-Reagan.txt',
 '1985-Reagan.txt',
 '1989-Bush.txt',
 '1993-Clinton.txt',
 '1997-Clinton.txt',
 '2001-Bush.txt',
 '2005-Bush.txt',
 '2009-Obama.txt',
 '2013-Obama.txt',
```

```
 '2017-Trump.txt',
 '2021-Biden.txt']
```

In [9]:
```
inaugural.words(fileids='1861-Lincoln.txt')[:20]
```

Out[9]:
```
['Fellow',
 '-',
 'Citizens',
 'of',
 'the',
 'United',
 'States',
 ':',
 'In',
 'compliance',
 'with',
 'a',
 'custom',
 'as',
 'old',
 'as',
 'the',
 'Government',
 'itself',
 ',']
```

In [1]:
```
from nltk.corpus import stopwords
```

In [4]:
```
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]
```

In [5]:
```
from nltk.corpus import stopwords

from nltk.tokenize import word_tokenize


example_sent = """This is a sample sentence,

                  showing off the stop words filtration."""


stop_words = set(stopwords.words('english'))


word_tokens = word_tokenize(example_sent)
# converts the words in word_tokens to lower case and then checks whether
#they are present in stop_words or not

filtered_sentence = [w for w in word_tokens if not w.lower() in stop_words]
#with no lower case conversion

filtered_sentence = []


for w in word_tokens:

    if w not in stop_words:

        filtered_sentence.append(w)


print(word_tokens)

print(filtered_sentence)
```

```
['This', 'is', 'a', 'sample', 'sentence', ',', 'showing', 'off', 'the', 'sto
p', 'words', 'filtration', '.']
['This', 'sample', 'sentence', ',', 'showing', 'stop', 'words', 'filtration',
'.']
```

In [6]:
```
from nltk.corpus import webtext
```

In [7]:
```
webtext.raw('grail.txt')
```

```
OR: You fight with the strength of many men, Sir Knight. [pause] I am Arth
ur, King of the Britons. [pause] I seek the finest and the bravest knights
in the land to join me in my court at Camelot. [pause] You have proved you
rself worthy.  Will you join me? [pause] You make me sad.  So be it.  Com
e, Patsy.\nBLACK KNIGHT: None shall pass.\nARTHUR: What?\nBLACK KNIGHT: No
ne shall pass.\nARTHUR: I have no quarrel with you, good Sir Knight, but I
must cross this bridge.\nBLACK KNIGHT: Then you shall die.\nARTHUR: I comm
and you, as King of the Britons, to stand aside!\nBLACK KNIGHT: I move for
no man.\nARTHUR: So be it!\nARTHUR and BLACK KNIGHT: Aaah!, hiyaah!, etc.
[ARTHUR chops the BLACK KNIGHT's left arm off] \nARTHUR: Now stand aside,
worthy adversary.\nBLACK KNIGHT: 'Tis but a scratch.\nARTHUR: A scratch?
Your arm's off!\nBLACK KNIGHT: No, it isn't.\nARTHUR: Well, what's that th
en?\nBLACK KNIGHT: I've had worse.\nARTHUR: You liar!\nBLACK KNIGHT: Come
on, you pansy! [clang] Huyah! [clang] Hiyaah! [clang] Aaaaaaaah! [ARTHUR c
hops the BLACK KNIGHT's right arm off] \nARTHUR: Victory is mine! [kneelin
g] We thank Thee Lord, that in Thy mer--\nBLACK KNIGHT: Hah! [clunk] Come
on then.\nARTHUR: What?\nBLACK KNIGHT: Have at you! [kick] \nARTHUR: Eh.
You are indeed brave, Sir Knight, but the fight is mine.\nBLACK KNIGHT: O
h, had enough, eh?\nARTHUR: Look, you stupid bastard.  You've got no arms
left.\nBLACK KNIGHT: Yes I have.\nARTHUR: Look!\nBLACK KNIGHT: Just a fles
```

In [1]:
```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\SAYEE\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
```

Out[1]:
```
True
```

In [2]:
```
from nltk.tokenize import sent_tokenize, word_tokenize
```

In [3]:
```
corpus = """Hellow welcome to the world of NLP.
This is the first step toward's learning NLP.
we will be learning NLP in depth."""
```

In [4]:
```
documents = sent_tokenize(corpus)
print(documents)
```

```
['Hellow welcome to the world of NLP.', "This is the first step toward's lear
ning NLP.", 'we will be learning NLP in depth.']
```

In [5]:
```
for doc in documents:
  print(word_tokenize(doc))
```

```
['Hellow', 'welcome', 'to', 'the', 'world', 'of', 'NLP', '.']
['This', 'is', 'the', 'first', 'step', 'toward', "'s", 'learning', 'NLP',
'.']
['we', 'will', 'be', 'learning', 'NLP', 'in', 'depth', '.']
```

In [6]:
```
from nltk.tokenize import word_tokenize
```

In [7]:
```
word_tokenize(corpus)
```

Out[7]:
```
['Hellow',
 'welcome',
 'to',
 'the',
 'world',
 'of',
 'NLP',
 '.',
 'This',
 'is',
 'the',
 'first',
 'step',
 'toward',
 "'s",
 'learning',
 'NLP',
 '.',
 'we',
 'will',
 'be',
 'learning',
 'NLP',
 'in',
 'depth',
 '.']
```

In [8]:
```
from nltk.tokenize import wordpunct_tokenize
```

In [9]:
```
wordpunct_tokenize(corpus)
```

Out[9]:
```
['Hellow',
 'welcome',
 'to',
 'the',
 'world',
 'of',
 'NLP',
 '.',
 'This',
 'is',
 'the',
 'first',
 'step',
 'toward',
 "'",
 's',
 'learning',
 'NLP',
 '.',
 'we',
 'will',
 'be',
 'learning',
 'NLP',
 'in',
 'depth',
 '.']
```

In [10]:
```
from nltk.tokenize import TreebankWordTokenizer
```

In [11]:
```
tokenizer = TreebankWordTokenizer()
```

In [12]:
```
tokenizer.tokenize(corpus)
```

Out[12]:
```
['Hellow',
 'welcome',
 'to',
 'the',
 'world',
 'of',
 'NLP.',
 'This',
 'is',
 'the',
 'first',
 'step',
 'toward',
 "'s",
 'learning',
 'NLP.',
 'we',
 'will',
 'be',
 'learning',
 'NLP',
 'in',
 'depth',
 '.']
```

In [13]:
```
words = ["eating","eaten","eats","writing","writes","programing","programs","h
```

In [14]:
```
from nltk.stem import PorterStemmer
stemming = PorterStemmer()
```

In [15]:
```
for word in words:
  print(word+"---->"+stemming.stem(word))
```

```
eating---->eat
eaten---->eaten
eats---->eat
writing---->write
writes---->write
programing---->program
programs---->program
history---->histori
finally---->final
finalized---->final
```

In [16]:
```
stemming.stem("congratulations") # not good
```

Out[16]:
```
'congratul'
```

In [17]:
```
stemming.stem("sitting") #good
```

Out[17]: 'sit'

In [18]:
```
from nltk.stem import RegexpStemmer
```

In [19]:
```
reg_stemmer = RegexpStemmer('ing$|s$|e$|able$', min=4)
```

In [20]:
```
reg_stemmer.stem("eating")
```

Out[20]: 'eat'

In [21]:
```
reg_stemmer.stem("ingeating")
```

Out[21]: 'ingeat'

In [22]:
```
from nltk.stem import SnowballStemmer
```

In [23]:
```
snowball = SnowballStemmer("english")
```

In [24]:
```
snowball.stem("history")
```

Out[24]: 'histori'

In [25]:
```
print("word: Fairly == "+"Porterstemmer -> "+stemming.stem("fairly")+". snowba
```

```
word: Fairly == Porterstemmer -> fairli. snowballstemmer -> fair
```

In [ ]:
```

```