Q.1 Tokenize the given text into sentences and words.

```
pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.7/dist-packages (3.2.5)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from nltk)
```

```
import nltk
from nltk.tokenize import sent_tokenize
from nltk.tokenize import word_tokenize
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

```
text = "The Department of Information Technology at Ramniranjan Jhunjhunwala College, Ghatkop
word_tokenize(text)
```

```
['The',
 'Department',
 'of',
 'Information',
 'Technology',
 'at',
 'Ramniranjan',
 'Jhunjhunwala',
 'College',
 ',',
 'Ghatkopar',
 ',',
 'Mumbai',
 'was',
 'established',
 'in',
 '2007',
 '.',
 'The',
 'Department',
 'offers',
 'both',
 'undergraduate',
 '(',
 'B.Sc',
 '.',
 'IT',
 ')',
 'and',
 'postgraduate',
 '(',
 'M.Sc',
```

```
 '.',
 'IT',
 ')',
 'programmes',
 '.',
 'The',
 'M.Sc',
 'IT',
 'programme',
 'was',
 'introduced',
 'in',
 'the',
 'year',
 '2016',
 '.']
```

```
sent_tokenize(text)
```

```
['The Department of Information Technology at Ramniranjan Jhunjhunwala College, Ghatkopa
 'The Department offers both undergraduate (B.Sc.',
 'IT) and postgraduate (M.Sc.',
 'IT) programmes.',
 'The M.Sc IT programme was introduced in the year 2016.']
```

Q.2 Pos tag each word to display its grammatical information

```
from nltk.chunk.regexp import tag_pattern2re_pattern
import matplotlib
matplotlib.use('Agg')
import nltk
nltk.download('averaged_perceptron_tagger')
from nltk.tokenize import word_tokenize
from nltk import pos_tag

def chunking(text, grammar):
    word_tokens = word_tokenize(text)

    # label words with part of speech
    word_pos = pos_tag(word_tokens)

    # create a chunk parser using grammar
    chunkParser = nltk.RegexpParser(grammar)

    # test it on the list of word tokens with tagged pos
    tree = chunkParser.parse(word_pos)

    for subtree in tree.subtrees():
        print(subtree)
```

```
sentence = 'the little yellow bird is flying in the sky'
grammar = "NP: {<DT>?<JJ>*<NN>}"
chunking(sentence, grammar)
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
(S
  (NP the/DT little/JJ yellow/JJ bird/NN)
  is/VBZ
  flying/VBG
  in/IN
  (NP the/DT sky/NN))
(NP the/DT little/JJ yellow/JJ bird/NN)
(NP the/DT sky/NN)
```

Q.3 Apply chunking to extract the following sentence from the given text. The Department offers both undergraduate (B.Sc. IT) and postgraduate (M.Sc. IT) programmes.

```
import nltk


locs = [('Omnicom', 'IN', 'New York'),
...         ('DDB Needham', 'IN', 'New York'),
...         ('Kaplan Thaler Group', 'IN', 'New York'),
...         ('BBDO South', 'IN', 'Atlanta'),
...         ('Georgia-Pacific', 'IN', 'Atlanta')]
query = [e1 for (e1, rel, e2) in locs if e2=='Atlanta']
print(query)

def ie_preprocess(document):
...    sentences = nltk.sent_tokenize(document)
...    sentences = [nltk.word_tokenize(sent) for sent in sentences]
...    sentences = [nltk.pos_tag(sent) for sent in sentences]

sentence = [("The", "Department"), ("offers", "both"), ("undergraduate", "(B.Sc. IT)"),
... ("and", "postgraduate"), ("(M.Sc. IT)", "programmes")]
grammar = "NP: {<DT>?<JJ>*<NN>}"
cp = nltk.RegexpParser(grammar)
result = cp.parse(sentence)
print(result)
```

```
['BBDO South', 'Georgia-Pacific']
(S
  The/Department
  offers/both
  undergraduate/(B.Sc. IT)
  and/postgraduate
  (M.Sc. IT)/programmes)
```

Q. 4 After chunking (Q. 3), POS tag the sentence and search for the information about programs B.Sc. IT and M.Sc. IT using POS taggers and Regular Expression.

```
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
True
```

```
from nltk.tokenize import word_tokenize
from nltk import pos_tag

# convert text into word_tokens with their tags
def pos_tagging(text):
  word_tokens = word_tokenize(text)
  return pos_tag(word_tokens)

pos_tagging('The Department offers both undergraduate (B.Sc. IT) and postgraduate (M.Sc. IT)
```

```
[('The', 'DT'),
 ('Department', 'NNP'),
 ('offers', 'VBZ'),
 ('both', 'DT'),
 ('undergraduate', 'NN'),
 ('(', '('),
 ('B.Sc', 'NNP'),
 ('.', '.'),
 ('IT', 'NNP'),
 (')', ')'),
 ('and', 'CC'),
 ('postgraduate', 'NN'),
 ('(', '('),
 ('M.Sc', 'NNP'),
 ('.', '.'),
 ('IT', 'NNP'),
 (')', ')'),
 ('programmes', 'NNS')]
```

Q.5 Display the parser tree for the Noun Phrase for the chunk derived in Q. 4

```
import matplotlib.pyplot as plt
import nltk
import string
import re
```

```
def ie_preprocess(document):
...    sentences = nltk.sent_tokenize(document) [1]
...    sentences = [nltk.word_tokenize(sent) for sent in sentences] [2]
...    sentences = [nltk.pos_tag(sent) for sent in sentences]


sentence = [("The", "Department"), ("offers", "both"), ("undergraduate", "(B.Sc. IT)"),
... ("and", "postgraduate"), ("(M.Sc. IT)", "programmes")]


grammar = "NP: {<DT>?<JJ>*<NN>}"
cp = nltk.RegexpParser(grammar)
result = cp.parse(sentence)
print (result)
```

```
     (S
       The/Department
       offers/both
       undergraduate/(B.Sc. IT)
       and/postgraduate
       (M.Sc. IT)/programmes)
```

Q.6 Get the Bog of words for the given text and display the word with its frequency.

```
import pandas as pd


from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer


text = "The Department of Information Technology at Ramniranjan Jhunjhunwala College, Ghatkop


CountVec = CountVectorizer(ngram_range=(1,1), stop_words='english')


Count_data = CountVec.fit_transform([text])


cv_dataframe=pd.DataFrame(Count_data.toarray(),columns=CountVec.get_feature_names())


print(cv_dataframe)
```

```
        2007  2016  college  department  ...  sc  technology  undergraduate  year
     0     1     1        1           2  ...   3           1              1     1

     [1 rows x 19 columns]
```

Q. 7 Remove the stop words from the BOW that is retrieved in Q. 6.

```
nltk.download('punkt')
```

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

```
from nltk.corpus import stopwords

nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

```
stop_words = set(stopwords.words("english"))
sentence = "The Department of Information Technology at Ramniranjan Jhunjhunwala College, Gha


words = nltk.word_tokenize(sentence)
without_stop_words = [word for word in words if not word in stop_words]


print(without_stop_words)
```

```
['The', 'Department', 'Information', 'Technology', 'Ramniranjan', 'Jhunjhunwala', 'Colle
```

Q.8 Stem the Words of Q. 7

```
import nltk
nltk.download('punkt')
```

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

```
from nltk.tokenize import word_tokenize
from nltk.stem.porter import PorterStemmer

text = 'The Department of Information Technology at Ramniranjan Jhunjhunwala College, Ghatkop

# Tokenize the string
tokens = word_tokenize(text)
print(tokens)
#=> ['fish', 'fishing', 'fishes', 'fisher', 'fished', 'fishy']

stemmer = PorterStemmer()
stems = [stemmer.stem(w) for w in tokens]
print(stems)
```

```
['The', 'Department', 'of', 'Information', 'Technology', 'at', 'Ramniranjan', 'Jhunjhunv
['the', 'depart', 'of', 'inform', 'technolog', 'at', 'ramniranjan', 'jhunjhunwala', 'col
```

*Q 9 .Find and display Lemma for the words that are retrieved in Q. 7 using lemmatization. *

```
import nltk
nltk.download('punkt')
from nltk.stem import WordNetLemmatizer
import nltk
nltk.download('wordnet')
```

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

```
# Define the sentence to be lemmatized
sentence = "The Department of Information Technology at Ramniranjan Jhunjhunwala College, Gha

# Tokenize: Split the sentence into words
word_list = nltk.word_tokenize(sentence)
print(word_list)
#> ['The', 'striped', 'bats', 'are', 'hanging', 'on', 'their', 'feet', 'for', 'best']

# Lemmatize list of words and join
lemmatized_output = ' '.join([lemmatizer.lemmatize(w) for w in word_list])
print(lemmatized_output)
#> The striped bat are hanging on their foot for best
```

```
['The', 'Department', 'of', 'Information', 'Technology', 'at', 'Ramniranjan', 'Jhunjhunw
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
```

** Q. 10 Find the synonym and antonym of words ‘establish’ and ‘introduce’.**

```
9 # Lemmatize list of words and join
```

```
import nltk
nltk.download('wordnet')
from nltk.corpus import wordnet
synonyms = []
antonyms = []

for syn in wordnet.synsets("establish"):
  for l in syn.lemmas():
    synonyms.append(l.name())
    if l.antonyms():
      antonyms.append(l.antonyms()[0].name())

print(set(synonyms))
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
{'demonstrate', 'base', 'ground', 'install', 'set_up', 'show', 'plant', 'establish', 'ir
{'abolish', 'disprove'}
```

```
print(set(antonyms))
```

```
{'abolish', 'disprove'}
```