Name: Siddharth Singh
Moodle Id: 20102176
Div: BE-C
Subject: NLP

Experiment No. 5

```
from nltk import *
from nltk import ngrams
nltk.download('wordnet')
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

```
print("\n============================")
print("POS tags")

text = """Hi Mr. Suresh. Everything is shining  about NLP in Python to understand 
How to installing nltk is always a big mystery less explained or properly documente

#Tokenize the text
text1 = nltk.word_tokenize(text)
print("Test sentence is:", text)
print("\nPOS List for tokens is ")
for token in text1:
  print (nltk.pos_tag([token]))
```

```
print("\n============================")
# let us use a lemmatizer
from nltk.stem import WordNetLemmatizer

# Init the Wordnet Lemmatizer
lemmatizer = WordNetLemmatizer()
print("text sentence is: ", text)
print("\nPOS list after lemmatization of tokens is ")
for token in text1:
  print(nltk.pos_tag([lemmatizer.lemmatize(token)]))
```

```
[('one', 'CD')]
[('greatest', 'JJS')]
[('language', 'NN')]
[('.', '.')]
[('How', 'WRB')]
[('to', 'TO')]
[('installing', 'VBG')]
[('nltk', 'NN')]
[('is', 'VBZ')]
[('always', 'RB')]
[('a', 'DT')]
[('big', 'JJ')]
[('mystery', 'NN')]
[('less', 'RBR')]
[('explained', 'VBD')]
```

```
[('or', 'CC')]
[('properly', 'RB')]
[('documented', 'VBN')]
[('.', '.')]

============================
text sentence is:  Hi Mr. Suresh. Everything is shining  about NLP in Python
How to installing nltk is always a big mystery less explained or properly do

POS list after lemmatization of tokens is
[('Hi', 'NN')]
[('Mr.', 'NNP')]
[('Suresh', 'NN')]
[('.', '.')]
[('Everything', 'NN')]
[('is', 'VBZ')]
[('shining', 'VBG')]
[('about', 'IN')]
[('NLP', 'NN')]
[('in', 'IN')]
[('Python', 'NN')]
[('to', 'TO')]
[('understand', 'NN')]
[('one', 'CD')]
[('greatest', 'JJS')]
[('language', 'NN')]
[('.', '.')]
[('How', 'WRB')]
[('to', 'TO')]
[('installing', 'VBG')]
[('nltk', 'NN')]
[('is', 'VBZ')]
[('always', 'RB')]
[('a', 'DT')]
[('big', 'JJ')]
[('mystery', 'NN')]
[('le', 'NN')]
[('explained', 'VBD')]
[('or', 'CC')]
[('properly', 'RB')]
[('documented', 'VBN')]
[('.', '.')]
```

```
# Filter insignificant words based on POS tags
'''Example insignificant words and their POS tags
a   DT
all PDT
an  DT
and CC
or  CC
that WDT
the  DT'''

# define function to remove insignificant POS tags (customised list)
def filter_insignificant(chunk, tag_suffixes =['DT', 'CC']):
  good = []
  for word, tag in chunk:
```

```
      ok = True
      for suffix in tag_suffixes:
        if tag.endswith(suffix):
          ok = False
          break
        if ok:
          good.append((word, tag))
    return good

# eliminate DT and CC POS tags
print("Significant words: \n",
      filter_insignificant ([('the', 'DT'), ('terrible', 'JJ'),('movie', 'NN')]))

print("\nSignificant words as per user defined POS tags")
# choosing tag suffixes to be eliminated
print ("Significant words user defined: \n",
filter_insignificant ([('your', 'PRPS'),('book', 'NN'), ('is', 'VBZ'), ('that', 'WI
tag_suffixes = [ 'PRP', 'PRP$', 'WDT']))
```

```
Significant words:
 [('terrible', 'JJ'), ('terrible', 'JJ'), ('movie', 'NN'), ('movie', 'NN')]

Significant words as per user defined POS tags
Significant words user defined:
 [('your', 'PRPS'), ('your', 'PRPS'), ('your', 'PRPS'), ('book', 'NN'), ('boo
```