```
#NLP
import nltk
nltk.download('punkt')
#Tokenizing
from nltk.tokenize import *
text="""A newspaper is the strongest medium for news. People are reading newspapers
for decades. It has a huge contribution to globalization. Right now because of easy
internet connection, people don’t read printed newspapers often. They read the online
version."""
print("Sample text : \n ",text,"\n")
sent_tokenized=sent_tokenize(text)
print("Tokenizing by sentence : \n",sent_tokenized,"\n")
word_tokenized=word_tokenize(text)
print("Tokenizing by word : \n ",word_tokenized,"\n")
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
Sample text :
  A newspaper is the strongest medium for news. People are reading newspapers
for decades. It has a huge contribution to globalization. Right now because of easy
internet connection, people don’t read printed newspapers often. They read the online
version.

Tokenizing by sentence :
 ['A newspaper is the strongest medium for news.', 'People are reading newspapers\nfor decades.', 'It has a huge contribution to gl

Tokenizing by word :
  ['A', 'newspaper', 'is', 'the', 'strongest', 'medium', 'for', 'news', '.', 'People', 'are', 'reading', 'newspapers', 'for', 'deca
```

```
#Filtering stop words
import nltk
nltk.download('stopwords')
from nltk.corpus import stopwords
from string import punctuation
stopwords=stopwords.words('english')
punctuation=list(punctuation)
print("After filtering the stop words and punctuation : ")
for word in word_tokenized:
  if word.casefold() not in stopwords and word.casefold() not in punctuation:
    print(word)
#new_list=[word for word in word_tokenized if word.casefold() not in stopwords and word not in punctuation]
#print(new_list,"\n")
```

```
After filtering the stop words and punctuation :
newspaper
strongest
medium
news
People
reading
newspapers
decades
huge
contribution
globalization
Right
easy
internet
connection
people
’
read
printed
newspapers
often
read
online
version
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
#Stemming
from nltk.stem import PorterStemmer
ps = PorterStemmer()
words = ["reading", "globalization", "Being","Went","gone","going"]
print("Given words : ",words)
stemm=[ps.stem(i) for i in words ]
print("After stemming : ",stemm,"\n")
```

```
Given words :  ['reading', 'globalization', 'Being', 'Went', 'gone', 'going']
After stemming :  ['read', 'global', 'be', 'went', 'gone', 'go']
```

```
#Lemmatization
from nltk.stem import WordNetLemmatizer
import nltk
nltk.download('wordnet')
nltk.download('omw-1.4')
lem= WordNetLemmatizer()
print("rocks :", lem.lemmatize("rocks"))
print("corpora :", lem.lemmatize("corpora"))
print("better :", lem.lemmatize("better"))
print("believes :", lem.lemmatize("believes"),"\n")
print("better :", lem.lemmatize("went",pos="a"))
print("better :", lem.lemmatize("went",pos="v"))
print("better :", lem.lemmatize("went",pos="n"),"\n")
```

```
    [nltk_data] Downloading package wordnet to /root/nltk_data...
    [nltk_data] Downloading package omw-1.4 to /root/nltk_data...
    rocks : rock
    corpora : corpus
    better : better
    believes : belief

    better : went
    better : go
    better : went
```

```
nltk.download('averaged_perceptron_tagger')
nltk.download('maxent_ne_chunker')
nltk.download('words')
from nltk import RegexpParser
from nltk.tree import *
#POS Tag
postag=nltk.pos_tag(word_tokenized)
print("POS tagging : \n")
for i in postag:
  print(i)
#Chunking
print("\n")
grammar = "NP: {<DT>?<JJ>*<NN>}"
chunker = RegexpParser(grammar)
output = chunker.parse(postag)
print("After Chunking:\n",output)
output.pretty_print()
```

```
  people/NNS
  don/VBP
  (NP ’/JJ t/NN)
  (NP read/NN)
  printed/VBD
  newspapers/NNS
  often/RB
  ./.
  They/PRP
  read/VBD
  (NP the/DT online/JJ version/NN)
  ./.)

   ____________________________________________________________________________________________________________________________
  |      |          |         |     |      |         |         |            |          |         |       |    |       |      |
  |      |          |         |     |      |         |         |            |          |         |       |    |       |      |
is/VBZ the/DT strongest/JJS for/IN ./. People/NNS are/VBP reading/VBG newspapers/NNS for/IN decades/NNS ./. It/PRP has/VBZ to/TO
```

```
#NER
Text = "The russian president Vladimir Putin is in the Kremlin"
Tokenize = nltk.word_tokenize(Text)
POS_tags = nltk.pos_tag(Tokenize)
NameEn = nltk.ne_chunk(POS_tags)
print(NameEn)
```

```
(S
  The/DT
  russian/JJ
  president/NN
  (PERSON Vladimir/NNP Putin/NNP)
  is/VBZ
  in/IN
  the/DT
  (FACILITY Kremlin/NNP))
```

```
#Word grouping
#bigram
print(list(nltk.bigrams(word_tokenized)),"\n")
#trigram
print(list(nltk.trigrams(word_tokenized)),"\n")
#n-gram
print(list(nltk.ngrams(word_tokenized,5)),"\n")
```

```
[('A', 'newspaper'), ('newspaper', 'is'), ('is', 'the'), ('the', 'strongest'), ('strongest', 'medium'), ('medium', 'for'), ('for',

[('A', 'newspaper', 'is'), ('newspaper', 'is', 'the'), ('is', 'the', 'strongest'), ('the', 'strongest', 'medium'), ('strongest', 'm

[('A', 'newspaper', 'is', 'the', 'strongest'), ('newspaper', 'is', 'the', 'strongest', 'medium'), ('is', 'the', 'strongest', 'mediu
```