Shri Ramdeobaba College of Engineering and Nagpur

Department of Computer Science and Engineering

Natural Language Processing Lab

Name : Shantanu Mane

Branch : CSE - AIML (VIth SEM)

Roll Num : E-63

Importing the Dependencies

```
import nltk
from nltk.corpus import wordnet as wn
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
```

SynSets

```
wn.synsets('motorcar')
```

```
[Synset('car.n.01')]
```

```
wn.synset('car.n.01').lemma_names()
```

```
['car', 'auto', 'automobile', 'machine', 'motorcar']
```

```
syn = wn.synsets('program')[0]
```

```
print(f"Synset Name : {syn.name()}\nSynset Meaning : {syn.definition()}\nSynset Example : {syn.examples()}")
```

```
Synset Name : plan.n.01
Synset Meaning : a series of steps to be carried out or goals to be accomplished
Synset Example : ['they drew up a six-step plan', 'they discussed plans for a new bond issue']
```

```
wn.synsets('banks')
```

```
[Synset('banks.n.01'),
 Synset('bank.n.01'),
 Synset('depository_financial_institution.n.01'),
 Synset('bank.n.03'),
 Synset('bank.n.04'),
 Synset('bank.n.05'),
 Synset('bank.n.06'),
 Synset('bank.n.07'),
 Synset('savings_bank.n.02'),
 Synset('bank.n.09'),
 Synset('bank.n.10'),
 Synset('bank.v.01'),
 Synset('bank.v.02'),
 Synset('bank.v.03'),
 Synset('bank.v.04'),
 Synset('bank.v.05'),
 Synset('deposit.v.02'),
 Synset('bank.v.07'),
 Synset('trust.v.01')]
```

```
wn.synset('bank.n.01').lemma_names()
```

```
['bank']
```

```
wn.synsets('bank', pos=wn.VERB)
```

```
[Synset('bank.v.01'),
 Synset('bank.v.02'),
 Synset('bank.v.03'),
 Synset('bank.v.04'),
 Synset('bank.v.05'),
 Synset('deposit.v.02'),
 Synset('bank.v.07'),
 Synset('trust.v.01')]
```

```
syn_bank = wn.synset('bank.n.01')
syn_banks = wn.synset('bank.n.03')

print(f"similarity : {syn_bank.wup_similarity(syn_banks)}")
```

```
similarity : 0.6153846153846154
```

```
wn.synsets('fool', pos=wn.NOUN)
```

```
[Synset('fool.n.01'), Synset('chump.n.01'), Synset('jester.n.01')]
```

```
syn_ship = wn.synsets('ship')[0]
syn_boat = wn.synsets('bank')[0]

print(f"similarity : {syn_boat.wup_similarity(syn_ship)}")
```

```
similarity : 0.35294117647058826
```

Part A :

```
text = """The term language corpus is used to mean a number of rather different things. It may refer simply to any collection of li

# Tokenize the text into sentences
sentences = nltk.sent_tokenize(text)

# Tokenize, remove stopwords, and tag each sentence
sentence_tokens = []
for sentence in sentences:
    tokens = word_tokenize(sentence)
    tokens = [token.lower() for token in tokens if token.isalpha()]
    tokens = [token for token in tokens if token not in stopwords.words('english')]
    tagged_tokens = nltk.pos_tag(tokens)
    sentence_tokens.append(tagged_tokens)

sentence_tokens
```

```
[[('term', 'NN'),
  ('language', 'NN'),
  ('corpus', 'NN'),
  ('used', 'VBN'),
  ('mean', 'JJ'),
  ('number', 'NN'),
  ('rather', 'RB'),
  ('different', 'JJ'),
  ('things', 'NNS')],
 [('may', 'MD'),
  ('refer', 'VB'),
  ('simply', 'RB'),
  ('collection', 'JJ'),
  ('linguistic', 'JJ'),
  ('data', 'NNS'),
  ('example', 'NN'),
  ('written', 'VBN'),
  ('spoken', 'VBN'),
  ('signed', 'VBN'),
  ('multimodal', 'CC'),
```

```
# Extract all the synsets of the words from the sentences
synsets = {}
for sentence in sentence_tokens:
    for word in sentence:
        if word[0] not in synsets:
            synsets[word[0]] = set(wn.synsets(word[0]))
```

```
synsets
```

```
{'term': {Synset('condition.n.07'),
  Synset('term.n.01'),
  Synset('term.n.02'),
  Synset('term.n.04'),
  Synset('term.n.05'),
  Synset('term.n.06'),
  Synset('term.v.01'),
  Synset('terminus.n.03')},
 'language': {Synset('language.n.01'),
  Synset('language.n.05'),
  Synset('linguistic_process.n.02'),
  Synset('lyric.n.01'),
  Synset('speech.n.02'),
  Synset('terminology.n.01')},
 'corpus': {Synset('corpus.n.02'),
  Synset('corpus.n.03'),
  Synset('principal.n.04')},
 'used': {Synset('exploited.s.02'),
  Synset('practice.v.04'),
  Synset('secondhand.s.02'),
```

Part B :

```
# similarity between first and last five words of the sentence

first_five = sentence_tokens[0][:5]
last_five = sentence_tokens[0][-5:]

print(f"First Five : {first_five}\nLast Five : {last_five}")
```

```
First Five : [('term', 'NN'), ('language', 'NN'), ('corpus', 'NN'), ('used', 'VBN'), ('mean', 'JJ')]
Last Five : [('mean', 'JJ'), ('number', 'NN'), ('rather', 'RB'), ('different', 'JJ'), ('things', 'NNS')]
```

```
# similarity between first and last five words of the sentence

for word1 in first_five:
    for word2 in last_five:
        if word1[0] in synsets and word2[0] in synsets:
            for syn1 in synsets[word1[0]]:
                for syn2 in synsets[word2[0]]:
                    print(f"Similarity between {word1[0]} and {word2[0]} : {syn1.wup_similarity(syn2)}")
```

```
Similarity between term and mean : 0.13333333333333333
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.2222222222222222
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.1111111111111111
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.1
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.125
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.15384615384615385
Similarity between term and mean : 0.14285714285714285
Similarity between term and mean : 0.16666666666666666
Similarity between term and mean : 0.4444444444444444
Similarity between term and mean : 0.16666666666666666
```

```python
# maximum similarity between two words in the sentence

max_similarity = 0
max_word1 = ""
max_word2 = ""

for word1 in first_five:
    for word2 in last_five:
        if word1[0] in synsets and word2[0] in synsets:
            for syn1 in synsets[word1[0]]:
                for syn2 in synsets[word2[0]]:
                    similarity = syn1.wup_similarity(syn2)
                    if similarity > max_similarity:
                        max_similarity = similarity
```

```
Maximum Similarity : 1.0
Word 1 : mean
Word 2 : mean
```