Name: Vinayak V Thayil

Roll No:AM.EN.U4CSE21161

```
import nltk
from nltk import word_tokenize, pos_tag, ne_chunk
from nltk.corpus import treebank
from nltk.tag import hmm
```

```
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download("maxent_ne_chunker")
nltk.download("words")
nltk.download("treebank")
```

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

```
# Step 1: Read a sentence
sentence = "At eight o'clock on Thursday morning Arthur didn't feel very good."
print(sentence)
```

```
At eight o'clock on Thursday morning Arthur didn't feel very good.
```

```
# Step 2: Tokenize the sentence
tokens = word_tokenize(sentence)
print(tokens)
```

```
['At', 'eight', "o'clock", 'on', 'Thursday', 'morning', 'Arthur', 'did', "n't", 'feel', 'very', 'good', '.']
```

```
# Step 3: Find bi-grams and tri-grams
bi_grams = list(nltk.bigrams(tokens))
tri_grams = list(nltk.trigrams(tokens))
print(bi_grams)
print(tri_grams)
```

```
[('At', 'eight'), ('eight', "o'clock"), ("o'clock", 'on'), ('on', 'Thursday'), ('Thursday', 'morning'), ('morning', 'Arthur'), ('Arthur', 'did'), ('did', "n't"), ("n't", 'feel
[('At', 'eight', "o'clock"), ('eight', "o'clock", 'on'), ("o'clock", 'on', 'Thursday'), ('on', 'Thursday', 'morning'), ('Thursday', 'morning', 'Arthur'), ('morning', 'Arthur',
```

```
# Step 4: Find POS tags of each token
pos_tags = pos_tag(tokens)
print(pos_tags)
```

```
[('At', 'IN'), ('eight', 'CD'), ("o'clock", 'NN'), ('on', 'IN'), ('Thursday', 'NNP'), ('morning', 'NN'), ('Arthur', 'NNP'), ('did', 'VBD'), ("n't", 'RB'), ('feel', 'VB'), ('ve
```

```
# Step 5: Identify named entities
named_entities = ne_chunk(pos_tags)
print(named_entities)
```

```
(S
  At/IN
  eight/CD
  o'clock/NN
  on/IN
  Thursday/NNP
  morning/NN
  (PERSON Arthur/NNP)
  did/VBD
  n't/RB
  feel/VB
  very/RB
  good/JJ
  ./.)
```

```
# Step 6: Print a sentence and corresponding tags from treebank
treebank_sentence = treebank.tagged_sents()[0]
print("Treebank:", treebank_sentence)
```

```
Treebank: [('Pierre', 'NNP'), ('Vinken', 'NNP'), (',', ','), ('61', 'CD'), ('years', 'NNS'), ('old', 'JJ'), (',', ','), ('will', 'MD'), ('join', 'VB'), ('the', 'DT'), ('board'
```

```
# Step 7: Count unique tags in the corpus
unique_tags = set(tag for word, tag in treebank.tagged_words())
num_unique_tags = len(unique_tags)
print(num_unique_tags)
```

```
46
```

```
# Step 8: Find the most commonly occurring tag in the corpus
tag_freq_dist = nltk.FreqDist(tag for word, tag in treebank.tagged_words())
most_common_tag = tag_freq_dist.most_common(1)[0][0]
print(most_common_tag)
```

```
NN
```

```
# Step 9: Find the tag most frequently assigned to the word "bank"
bank_tags = [tag for word, tag in treebank.tagged_words() if word.lower() == "bank"]
most_common_bank_tag = nltk.FreqDist(bank_tags).most_common(1)[0][0]
print(most_common_bank_tag)
```

```
NN
```

```
# Step 10: Implement an HMM POS tagger in Python
train_data = treebank.tagged_sents()[:3000]
hmm_tagger = hmm.HiddenMarkovModelTagger.train(train_data)
hmm_tagger.tag(tokens)
```

```
[('At', 'IN'),
 ('eight', 'CD'),
 ("o'clock", 'NNS'),
 ('on', 'IN'),
 ('Thursday', 'NNP'),
 ('morning', 'NNP'),
 ('Arthur', 'NNP'),
 ('did', 'VBD'),
 ("n't", 'RB'),
 ('feel', 'VBP'),
 ('very', 'RB'),
 ('good', 'JJ'),
 ('.', '.')]
```