Name: Onasvee Banarse

Class: BE Computer-1

Roll No: 09

Problem Statement: Perform tokenization (Whitespace, Punctuation-based, Treebank, Tweet, MWE) using NLTK library. Use porter stemmer and snowball stemmer for stemming. Use any technique for lemmatization.

Sample Sentences

In [1]:

```
sentence1 = "It's true, Ms. Martha Jones! #Truth"
sentence2 = "I played the play playfully as the players were playing in the play wi
```

Tokenization

In [3]:

```
import nltk
from nltk.tokenize import (
    word_tokenize,
    wordpunct_tokenize,
    TreebankWordTokenizer,
    TweetTokenizer,
    MWETokenizer
)

print(f'Whitespace tokenization = {sentence1.split()}')

print(f'Punctuation-based tokenization = {wordpunct_tokenize(sentence1)}')

tokenizer = MWETokenizer()
tokenizer.add_mwe(('Martha', 'Jones'))
print(f'Multi-word expression (MWE) tokenization = {tokenizer.tokenize(word_tokeniz

tokenizer = TweetTokenizer()
print(f'Tweet-rules based tokenization = {tokenizer.tokenize(sentence1)}')

tokenizer = TreebankWordTokenizer()
print(f'Default/Treebank tokenization = {tokenizer.tokenize(sentence1)}')
```

```
Whitespace tokenization = ["It's", 'true,', 'Ms.', 'Martha', 'Jones!', '#Truth']
Punctuation-based tokenization = ['It', "'", 's', 'true', ',', 'Ms', '.', 'Marth
a', 'Jones', '!', '#', 'Truth']
Multi-word expression (MWE) tokenization = ['It', "'s", 'true', ',', 'Ms.', 'Marth
a_Jones', '!', '#', 'Truth']
Tweet-rules based tokenization = ["It's", 'true', ',', 'Ms', '.', 'Martha', 'Jone
s', '!', '#Truth']
Default/Treebank tokenization = ['It', "'s", 'true', ',', 'Ms.', 'Martha', 'Jone
s', '!', '#', 'Truth']
```

In [4]:

```
from nltk import word_tokenize, sent_tokenize
```

```
print('Tokenized words:', word_tokenize(sentence1))
print('\nTokenized sentences:', sent_tokenize(sentence1))
```

```
Tokenized words: ['It', "'s", 'true', ',', 'Ms.', 'Martha', 'Jones', '!', '#', 'Tr
uth']

Tokenized sentences: ["It's true, Ms. Martha Jones!", '#Truth']
```

Stemming

PorterStemmer

In [5]:

```
from nltk.stem import PorterStemmer

stemmer = PorterStemmer()

#list of tokenized words
token = word_tokenize(sentence2)

#stem's of each word
stem_words = [stemmer.stem(word) for word in token]

#print stemming results
for e1, e2 in zip(token, stem_words):
    print(e1.ljust(13), '-->', '\t', e2)
```

```
I             -->        i
played        -->        play
the           -->        the
play          -->        play
playfully     -->        play
as            -->        as
the           -->        the
players       -->        player
were          -->        were
playing       -->        play
in            -->        in
the           -->        the
play          -->        play
with          -->        with
playfullness  -->        playful
```

SnowballStemmer

In [6]:

```
from nltk.stem.snowball import SnowballStemmer

#the stemmer requires a language parameter
snow_stemmer = SnowballStemmer(language='english')

#list of tokenized words
token = word_tokenize(sentence2)

#stem's of each word
stem_words = [snow_stemmer.stem(word) for word in token]

#print stemming results
for e1, e2 in zip(token, stem_words):
    print(e1.ljust(13), '-->', '\t', e2)
```

```
I             -->        i
played        -->        play
the           -->        the
play          -->        play
playfully     -->        play
as            -->        as
the           -->        the
players       -->        player
were          -->        were
playing       -->        play
in            -->        in
the           -->        the
play          -->        play
with          -->        with
playfullness  -->        playful
```

Lemmatization

In [10]:

```python
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()

list1 = ['kites', 'babies', 'dogs', 'flying', 'smiling',
         'driving', 'died', 'tried', 'feet']

lemmatized_output = [lemmatizer.lemmatize(word) for word in list1]

#print stemming results
for e1, e2 in zip(list1, lemmatized_output):
    print(e1.ljust(13), '-->', '\t', e2)
```

```
kites         -->        kite
babies        -->        baby
dogs          -->        dog
flying        -->        flying
smiling       -->        smiling
driving       -->        driving
died          -->        died
tried         -->        tried
feet          -->        foot
```