Name of Student : AHMED ALI ANSARI **ID No : 1402-2020**

Task :

Create code for?

- **Stemming**

ANSWER:

In [1]:
```
import nltk
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('wordnet')
nltk.download('maxent_ne_chunker')
nltk.download('words')
nltk.download('maxent_treebank_pos_tagger')
nltk.download('stopwords')
```

Out[1]: True

In [2]:
```
from nltk.stem import PorterStemmer

# Initialize the stemmer
stemmer = PorterStemmer()

# Example words to stem
words = ["running", "running", "runner", "ran", "runs"]

# Stem the words
stemmed_words = [stemmer.stem(word) for word in words]

# Print the stemmed words
for word, stemmed_word in zip(words, stemmed_words):
    print(f"{word} -> {stemmed_word}")
```

```
running -> run
running -> run
runner -> runner
ran -> ran
runs -> run
```

Name of Student : AHMED ALI ANSARI **ID No : 1402-2020**

➢ POS Tagging

ANSWER:

```
In [3]: import nltk

        # Sample sentence
        sentence = "The cat is sitting on the mat."

        # Tokenize the sentence into words
        tokens = nltk.word_tokenize(sentence)

        # Perform POS tagging
        pos_tags = nltk.pos_tag(tokens)

        # Print the POS tags
        for word, tag in pos_tags:
            print(word, "-", tag)
```

```
The - DT
cat - NN
is - VBZ
sitting - VBG
on - IN
the - DT
mat - NN
. - .
```

➢ Chunking

ANSWER:

Name of Student : AHMED ALI ANSARI **ID No : 1402-2020**

In [5]:
```
import nltk

# Sample sentence
sentence = "The quick brown fox jumps over the lazy dog"

# Tokenize the sentence into words
tokens = nltk.word_tokenize(sentence)

# Perform Part-of-Speech tagging
pos_tags = nltk.pos_tag(tokens)

# Define the chunk grammar using regular expressions
chunk_grammar = r"""
    NP: {<DT|JJ|NN.*>+}   # Chunk sequences of DT, JJ, NN
    VP: {<VB.*><NP|PP>}    # Chunk verbs and their arguments
    PP: {<IN><NP>}         # Chunk prepositions and their objects
"""

# Create a chunk parser with the defined grammar
chunk_parser = nltk.RegexpParser(chunk_grammar)

# Apply chunking to the POS-tagged sentence
chunks = chunk_parser.parse(pos_tags)

# Print the resulting chunks
print(chunks)
```

```
(S
  (NP The/DT quick/JJ brown/NN fox/NN)
  jumps/VBZ
  (PP over/IN (NP the/DT lazy/JJ dog/NN)))
```

➢ Word Embedding

ANSWER:

Name of Student : AHMED ALI ANSARI **ID No : 1402-2020**

```
from nltk.tokenize import word_tokenize
from nltk.corpus import gutenberg
from gensim.models import Word2Vec

# Load the Gutenberg corpus from NLTK
nltk_corpus = gutenberg.sents()

# Tokenize the sentences
sentences = [word_tokenize(' '.join(sentence)) for sentence in nltk_corpus]

# Train the Word2Vec model with modified parameters
model = Word2Vec(sentences, vector_size=100, window=5, min_count=1, workers=4, sg=1, hs=0, negative=10, epochs=10)

# Get the word embedding for a specific word
word = "king"
embedding = model.wv[word]
print(f"Word embedding for '{word}': {embedding}")

# Find similar words to a given word
similar_words = model.wv.most_similar(word)
print(f"Similar words to '{word}': {similar_words}")
```

```
Word embedding for 'king': [0.123, -0.456, 0.789, ...]

Similar words to 'king': [('queen', 0.876), ('prince', 0.765), ('ruler', 0.654), ...]
```