### Program Aim:

This lab demonstrates various NLP tasks such as stop word elimination, stemming, lemmatization, POS tagging, and lexical analysis using the NLTK library in Python.

#### Full Library Imports:

* nltk.corpus.stopwords
* nltk.tokenize.word\_tokenize
* nltk.stem.PorterStemmer
* nltk.stem.WordNetLemmatizer
* nltk.tokenize.sent\_tokenize
* nltk.pos\_tag

### Step 1: **Stop Word Elimination**

Stop words are commonly used words (like “is”, “the”, “in”) that do not contribute significantly to the meaning of a sentence and are usually removed to reduce noise in text processing tasks.

#### Libraries Used:

| **Library** | **Description** |
| --- | --- |
| stopwords | Object: Contains predefined stop words in various languages. The method words() accepts the language as an argument (e.g., ‘english’) to return the stop words. |
| word\_tokenize | Method: Tokenizes a string into individual words. Accepts a sentence (string) as input. |

#### Code:

import nltk  
from nltk.corpus import stopwords  
from nltk.tokenize import word\_tokenize  
  
nltk.download('punkt')  
nltk.download('stopwords')  
  
def remove\_stopwords(text):  
 stop\_words = set(stopwords.words('english'))  
 word\_tokens = word\_tokenize(text)  
 filtered\_text = [word for word in word\_tokens if word.lower() not in stop\_words]  
 return ' '.join(filtered\_text)  
  
text = "This is an example sentence demonstrating stop word removal."  
filtered\_text = remove\_stopwords(text)  
print(filtered\_text)

#### Sample Input:

"This is an example sentence demonstrating stop word removal."

#### Sample Output:

example  
sentence  
demonstrating  
stop  
word  
removal

### Step 2: **Stemming**

Stemming is the process of reducing words to their base or root form by removing suffixes.

#### Libraries Used:

| **Library** | **Description** |
| --- | --- |
| PorterStemmer | Object: Performs stemming using the Porter algorithm. No arguments are needed for initialization. |
| word\_tokenize | Method: Tokenizes a string into individual words. Accepts a sentence as input. |

#### Code:

import nltk  
from nltk.stem import PorterStemmer  
from nltk.tokenize import word\_tokenize  
  
text = input("Enter a sentence for stemming: ")  
words = word\_tokenize(text)  
stemmer = PorterStemmer()  
stemmed\_words = [stemmer.stem(word) for word in words]  
  
print("\nAfter Stemming:")  
print(stemmed\_words)

#### Sample Input:

"Running faster is the best option."

#### Sample Output:

['run',  
 'faster',  
 'is',  
 'the',  
 'best',  
 'option']

### Step 3: **Lemmatization**

Lemmatization reduces words to their base or dictionary form, known as the lemma, considering the word’s context and part-of-speech.

#### Libraries Used:

| **Library** | **Description** |
| --- | --- |
| WordNetLemmatizer | Object: Lemmatizes words based on their dictionary form. No arguments required for initialization. |
| word\_tokenize | Method: Tokenizes a string into words. Accepts a sentence as input. |

#### Code:

import nltk  
from nltk.stem import WordNetLemmatizer  
  
text = "This is a sample text. It contains some words that we can use for lemmatization."  
tokens = nltk.word\_tokenize(text)  
lemmatizer = WordNetLemmatizer()  
  
for token in tokens:  
 lemma = lemmatizer.lemmatize(token)  
 print(token, "-->", lemma)

#### Sample Input:

"The leaves are falling"

#### Sample Output:

The --> The  
leaves --> leaf  
are --> are  
falling --> falling

### Step 4: **Part-of-Speech (POS) Tagging**

POS tagging assigns grammatical tags to each word in a sentence, identifying whether a word is a noun, verb, adjective, etc.

#### Libraries Used:

| **Library** | **Description** |
| --- | --- |
| word\_tokenize | Method: Tokenizes a sentence into words. Accepts a string as input. |
| pos\_tag | Method: Tags words with their corresponding part of speech. Takes a list of word tokens as input. |

#### Code:

import nltk  
from nltk.tokenize import word\_tokenize  
  
nltk.download('punkt')  
nltk.download('averaged\_perceptron\_tagger')  
  
text1 = "children are innocent."  
tokens = word\_tokenize(text1)  
tagged\_words = nltk.pos\_tag(tokens)  
print(tagged\_words)  
  
text2 = "Visiting aunts can be a nuisance"  
tokens = word\_tokenize(text2)  
tagged\_words = nltk.pos\_tag(tokens)  
print(tagged\_words)

#### Sample Input:

"Visiting aunts can be a nuisance"

#### Sample Output:

[('Visiting', 'VBG'),  
 ('aunts', 'NNS'),  
 ('can', 'MD'),  
 ('be', 'VB'),  
 ('a', 'DT'),  
 ('nuisance', 'NN')]

### Step 5: **Lexical Analysis**

Lexical analysis involves breaking text into sentences and words, followed by assigning part-of-speech tags to the tokens.

#### Libraries Used:

| **Library** | **Description** |
| --- | --- |
| sent\_tokenize | Method: Tokenizes text into sentences. Accepts a string as input. |
| word\_tokenize | Method: Tokenizes text into words. Accepts a string or sentence as input. |
| pos\_tag | Method: Assigns part-of-speech tags to words. Takes word tokens as input. |

#### Code:

import nltk  
from nltk.tokenize import word\_tokenize, sent\_tokenize  
  
nltk.download('punkt')  
  
text = "This is a sample text for lexical analysis using NLTK."  
  
# Sentence Tokenization  
sentences = sent\_tokenize(text)  
print("Sentences:", sentences)  
  
# Word Tokenization  
words = word\_tokenize(text)  
print("Words:", words)  
  
# Part-of-Speech Tagging  
nltk.download('averaged\_perceptron\_tagger')  
tagged\_words = nltk.pos\_tag(words)  
print("Tagged Words:", tagged\_words)

#### Sample Input:

"This is a sample text for lexical analysis using NLTK."

#### Sample Output:

Sentences:  
['This is a sample text for lexical analysis using NLTK.']  
  
Words:  
['This',  
 'is',  
 'a',  
 'sample',  
 'text',  
 'for',  
 'lexical',  
 'analysis',  
 'using',  
 'NLTK',  
 '.']  
  
Tagged Words:  
[('This', 'DT'),  
 ('is', 'VBZ'),  
 ('a', 'DT'),  
 ('sample', 'NN'),  
 ('text', 'NN'),  
 ('for', 'IN'),  
 ('lexical', 'JJ'),  
 ('analysis', 'NN'),  
 ('using', 'VBG'),  
 ('NLTK', 'NNP'),  
 ('.', '.')]