**Stop word elimination:**

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)

**Stemming:**

import nltk

from nltk.stem import PorterStemmer

from nltk.tokenize import word\_tokenize

# Input from the user

text = input("Enter a sentence for stemming: ")

# Tokenization

words = word\_tokenize(text)

# Stemming

stemmer = PorterStemmer()

stemmed\_words = [stemmer.stem(word) for word in words]

print("\nAfter Stemming:")

print(stemmed\_words)

**Lemmatization:**

import nltk

from nltk.stem import WordNetLemmatizer

# Define a text string

text = "This is a sample text. It contains some words that we can use for lemmatization."

# Tokenize the text into individual words

tokens = nltk.word\_tokenize(text)

# Create a WordNetLemmatizer object

lemmatizer = WordNetLemmatizer()

# Lemmatize each word and print the result

for token in tokens:

lemma = lemmatizer.lemmatize(token)

print(token, "-->", lemma)

**POS Tagging:**

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)

print()

text2 = "Visiting aunts can be a nuisance"

tokens = nltk.word\_tokenize(text2)

tagged\_words = nltk.pos\_tag(tokens)

print(tagged\_words)

print()

**Lexical Analysis:**

import nltk

from nltk.tokenize import word\_tokenize, sent\_tokenize

# Download required resources if not already present

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)