program 3. Download Wikipedia's page on open source and convert the text to its native forms. Try it with various stemming and lemmatization modules. Use Python's timer module to measure their performance.

Objective: This program aims to analyze the efficiency of stemming and lemmatization in NLP by:

Downloading the Wikipedia summary of "Open Source."

Applying text preprocessing techniques such as stemming and lemmatization to convert words to their root forms.

Comparing performance by measuring execution time using Python's time module.

Methodology

Fetch Wikipedia Text: Retrieve the "Open Source" summary using the Wikipedia API.

Tokenize Text: Use spaCy for word segmentation.

Apply Stemming : Use PorterStemmer from nltk to stem words.

Apply Lemmatization: Use spaCy to lemmatize words.

Measure Execution Time: Use time module to compare performance of both techniques.

Display Results: Print sample outputs and execution times.

import libraries

```
!pip install wikipedia-api
```

```
Requirement already satisfied: wikipedia-api in /usr/local/lib/python3.11/dist-packages (0.8.1)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from wikipedia-api) (2.32.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->wikipedia-api) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests->wikipedia-api) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->wikipedia-api) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests->wikipedia-api) (2025.1.31)
```

```
import nltk
nltk.download('punkt')
```

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

```
nltk.download()
```

```
|   Package stopwords is already up-to-date!
| Downloading package subjectivity to /root/nltk_data...
|   Package subjectivity is already up-to-date!
| Downloading package swadesh to /root/nltk_data...
|   Package swadesh is already up-to-date!
| Downloading package switchboard to /root/nltk_data...
|   Package switchboard is already up-to-date!
| Downloading package tagsets to /root/nltk_data...
|   Package tagsets is already up-to-date!
| Downloading package tagsets_json to /root/nltk_data...
|   Package tagsets_json is already up-to-date!
| Downloading package timit to /root/nltk_data...
|   Package timit is already up-to-date!
| Downloading package toolbox to /root/nltk_data...
|   Package toolbox is already up-to-date!
| Downloading package treebank to /root/nltk_data...
|   Package treebank is already up-to-date!
| Downloading package twitter_samples to /root/nltk_data...
|   Package twitter_samples is already up-to-date!
| Downloading package udhr to /root/nltk_data...
|   Package udhr is already up-to-date!
| Downloading package udhr2 to /root/nltk_data...
|   Package udhr2 is already up-to-date!
| Downloading package unicode_samples to /root/nltk_data...
|   Package unicode_samples is already up-to-date!
| Downloading package universal_tagset to /root/nltk_data...
|   Package universal_tagset is already up-to-date!
| Downloading package universal_treebanks_v20 to
|     /root/nltk_data...
|   Package universal_treebanks_v20 is already up-to-date!
| Downloading package vader_lexicon to /root/nltk_data...
|   Package vader_lexicon is already up-to-date!
| Downloading package verbnet to /root/nltk_data...
|   Package verbnet is already up-to-date!
| Downloading package verbnet3 to /root/nltk_data...
|   Package verbnet3 is already up-to-date!
| Downloading package webtext to /root/nltk_data...
|   Package webtext is already up-to-date!
| Downloading package wmt15_eval to /root/nltk_data...
|   Package wmt15_eval is already up-to-date!
| Downloading package word2vec_sample to /root/nltk_data...
|   Package word2vec_sample is already up-to-date!
| Downloading package wordnet to /root/nltk_data...
|   Package wordnet is already up-to-date!
| Downloading package wordnet2021 to /root/nltk_data...
|   Package wordnet2021 is already up-to-date!
| Downloading package wordnet2022 to /root/nltk_data...
|   Package wordnet2022 is already up-to-date!
| Downloading package wordnet31 to /root/nltk_data...
|   Package wordnet31 is already up-to-date!
| Downloading package wordnet_ic to /root/nltk_data...
|   Package wordnet_ic is already up-to-date!
| Downloading package words to /root/nltk_data...
|   Package words is already up-to-date!
| Downloading package ycoe to /root/nltk_data...
|   Package ycoe is already up-to-date!
|
Done downloading collection all
```

```
import time  # Timer module for performance measurement
```

```
import wikipediaapi  # Wikipedia API to fetch text
import spacy  # NLP library for lemmatization
from nltk.stem import PorterStemmer  # Stemming module from nltk
from nltk.tokenize import word_tokenize  # Tokenization module
```

1. Fetch Wikipedia Content

```
# Step 1: Fetch Wikipedia content
def get_wikipedia_text(page_title):
    wiki_wiki = wikipediaapi.Wikipedia(user_agent="MyNLPProject/1.0", language="en")
    page = wiki_wiki.page(page_title)
    return page.summary if page.exists() else ""

text = get_wikipedia_text("Keshav_Memorial_Institute_of_Technology")
```

This function retrieves the summary of the Wikipedia page "Open Source" using wikipediaapi.

If the page exists, it returns the summary text.

2. Tokenization

```
# Tokenize the text
tokens = word_tokenize(text)
```

Splits the text into individual words (tokens), making it easier to process.

3. Apply Stemming

```
# Step 2: Apply Stemming
stemmer = PorterStemmer()

start_stem = time.time()  # Start timer
stemmed_words = [stemmer.stem(word) for word in tokens]
end_stem = time.time()  # End timer
```

Uses PorterStemmer from nltk to convert words into their stemmed form.

Example: "running" → "run", "better" → "bet".

4. Apply Lemmatization

```
# Step 3: Apply Lemmatization
nlp = spacy.load("en_core_web_sm")

start_lem = time.time()  # Start timer
doc = nlp(" ".join(tokens))
lemmatized_words = [token.lemma_ for token in doc]
end_lem = time.time()  # End timer
```

Uses spaCy to perform lemmatization, which provides proper root words.

Example: "running" → "run", "better" → "good".

The time module calculates execution times for stemming and lemmatization.

Used to compare performance differences between the two techniques.

5. Display Results and Performance Comparison

```
# Step 4: Display Results
print("Original Text Sample:", tokens[:10])
print("Stemmed Words:", stemmed_words[:10])
print("Lemmatized Words:", lemmatized_words[:10])

# Step 5: Performance Comparison
print("\nPerformance Analysis:")
print(f"Stemming Execution Time: {end_stem - start_stem:.5f} seconds")
print(f"Lemmatization Execution Time: {end_lem - start_lem:.5f} seconds")
```

```
Original Text Sample: ['Keshav', 'Memorial', 'Institute', 'of', 'Technology', 'is', 'a', 'private', 'engineering', 'college']
Stemmed Words: ['keshav', 'memori', 'institut', 'of', 'technolog', 'is', 'a', 'privat', 'engin', 'colleg']
Lemmatized Words: ['Keshav', 'Memorial', 'Institute', 'of', 'Technology', 'be', 'a', 'private', 'engineering', 'college']

Performance Analysis:
Stemming Execution Time: 0.00114 seconds
Lemmatization Execution Time: 0.02289 seconds
```