Instructions for Running the Program

This program generates random sentences based on bigrams (pairs of consecutive words) extracted from text files stored in a folder named **"data"**. Follow these steps to run the program successfully:

1. Install Required Libraries:

Before running the script, ensure you have the required Python libraries installed. Run:

pip install nltk

The program uses **NLTK (Natural Language Toolkit)** for tokenization and bigram analysis.

2. Prepare Training Data:

- Create a folder named **"data"** in the same directory as the script.
- Place **plain text (.txt) files** inside the "data" folder.
- Ensure the text files contain meaningful sentences.

Example folder structure:

/your_project_directory

  /data

    file1.txt

    file2.txt

  bigram_sentence_generator.py # (this script)

3. Run the Script

Execute the script in a Python environment:

python bigram_sentence_generator.py

4. Understanding the Code Workflow:

- The script reads all .txt files from the "data" folder.
- It **tokenizes** words and removes unnecessary punctuation (except full stops).
- It creates **bigrams (word pairs)** and calculates their frequency.
- The script selects the **top 3 most frequent bigrams** for each word.
- When given a **starting word**, the script generates a sentence by randomly selecting a likely next word.

5. Generating a Sentence:

Modify this line at the end of the script to start with any word:

generate_sentence('Asia', 10)

Here:

- 'Asia' is the **starting word**.
- 10 is the **number of words** in the generated sentence.

Example Output (if data is available):

Asia is a beautiful country with rich culture

If the word is missing from the dataset:

'Asia' not found in training data.

6. Debugging Tips:

- **Ensure the "data" folder exists** and contains readable .txt files.
- **Check for missing dependencies** (ModuleNotFoundError: No module named 'nltk' → Install nltk).
- **Download NLTK tokenization models** by running this once:

import nltk

nltk.download('punkt')

- **If no words are generated**, verify that the text files contain meaningful data.

7. Modifications & Enhancements:

- Change the **starting word** for different results.
- Increase **num_words** to generate longer sentences.
- Adjust topk=3 to store more bigrams per word.
- Modify random.choice(next_words) to use a weighted probability selection.