**🔍 Title: Word Prediction Model using n-grams with the Brown Corpus (NLTK)**

**Purpose of the Code**

The main objectives of this script are to:

• Clean a text corpus (the Brown corpus),  
• Generate n-grams (groups of n consecutive words),  
• Calculate their frequency of occurrence,  
• And finally, predict the most probable next word based on a given (n-1)-gram input.

⚙️ **Step-by-Step Description**

1. **Download and Load the Corpus**

nltk.download('brown')

nltk.download('punkt')

**Explanation:**  
This step downloads the Brown corpus and the Punkt tokenizer from the NLTK library, which are necessary for text processing and tokenization.

1. **Data Preprocessing**

def preprocess\_brown():

**Purpose:**  
To clean and tokenize the Brown corpus.

**Explanation:**  
This function loads the full Brown corpus, converts all words to lowercase to standardize them, and tokenizes the text using NLTK’s word\_tokenize() method.

**Returns:**  
A list of lowercase word tokens from the Brown corpus.

1. **n-gram Generation**

def get\_ngrams\_freq(words, n):

**Purpose:**  
To generate a frequency table of all n-grams from a list of words.

**Explanation:**  
This function slides a window of size *n* over the list of words to form n-grams. It counts the occurrences of each unique n-gram using Python’s Counter and returns the results as a pandas DataFrame.

**Parameters:**

* words (list): List of word tokens.
* n (int): Size of the n-grams.

**Returns:**  
A DataFrame with columns:

* 'ngram': Tuple of n words,
* 'freq': Frequency of occurrence.

1. **Next Word Prediction**

def predict\_next\_word(words, ngram\_df, input\_seq, n, k):

**Purpose:**  
To predict the most likely next word based on a given (n-1)-gram input.

**Explanation:**  
This function searches the ngram\_df for all n-grams that start with the given (n-1)-gram (input\_seq). It returns the top k most frequent next words (i.e., the final word in the matched n-grams) sorted by frequency.

**Parameters:**

* words (list): The original tokenized word list (not directly used in prediction).
* ngram\_df (DataFrame): A DataFrame containing n-grams and their frequencies.
* input\_seq (tuple): A sequence of (n-1) words.
* n (int): The n in n-gram.
* k (int): Number of top predictions to return.

**Returns:**  
A list of the top k most probable next words.

1. **Program Execution**

if \_\_name\_\_ == "\_\_main\_\_":

**Purpose:**  
To execute the main program logic when the script is run directly.

**Explanation:**  
Typically used to call the preprocessing function, generate n-grams, and test predictions with sample input. This section ensures the script can also be imported as a module without executing immediately.

