1. **Question Set #1**
    a. What do you consider a token for this task and why?
    In this task, a token refers to each individual letter and punctuation mark. This approach is chosen because the language model operates at the letter level, treating each letter as a separate unit of analysis. Including punctuation marks is essential as they can significantly impact the meaning and structure of words, especially in languages like French where diacritics and punctuation play a crucial role. Preprocessing steps involve removing any leading or trailing whitespaces, converting all letters to lowercase to ensure uniformity, and handling punctuation marks appropriately within words.

    b. What technique do you decide to use for out of vocabulary (OOV) words and why?
    For handling out of vocabulary (OOV) words, Laplace Smoothing is employed. OOV words refer to letter combinations or sequences that are not observed in the training data. Laplace Smoothing adds a small pseudo-count to every possible letter combination, ensuring that no combination has a zero probability. Laplace Smoothing is chosen because it provides a straightforward and effective way to handle unseen letter combinations, preventing the model from assigning zero probabilities to unseen sequences.

    c. Can the letter bigram model be implemented without any kind of smoothing? If not, use add-one smoothing. Is this kind of smoothing appropriate or do you need better algorithms? Why (not)?
    Implementing the letter bigram model without any smoothing is not possible, especially in practical applications where the training data may not cover all possible letter combinations. Without smoothing, the model would assign zero probabilities to unseen bigrams, leading to inaccuracies during language prediction. Add-one smoothing, also known as Laplace smoothing, is a suitable choice in this context. It involves adding one count to each possible bigram, effectively handling unseen combinations by ensuring that no bigram has a zero probability. While add-one smoothing is a simple and commonly used technique, it may not be optimal for all scenarios, particularly when dealing with large vocabularies or sparse data.

2. **Question Set #2**
    a. What do you consider as a word for this task and why?
    In this task, a word is defined as a sequence of alphanumeric characters representing a meaningful unit of language. Punctuation marks are excluded from word tokens as they are not considered part of the words themselves but rather separators or modifiers. Preprocessing steps include tokenizing the text into individual words, removing any leading or trailing whitespaces, and applying lemmatization or stemming to normalize word forms.

    b. What technique do you decide to use for out of vocabulary (OOV) words and why?
    Laplace Smoothing is chosen for handling out of vocabulary (OOV) words in the word bigram model. OOV words refer to word sequences or combinations not observed in the

training data. Laplace Smoothing assigns a small probability to unseen word bigrams, preventing the model from encountering zero probabilities for unseen sequences. Laplace Smoothing is selected due to its simplicity and effectiveness in handling unseen word combinations. However, alternative techniques such as Good-Turing smoothing or Katz smoothing could be explored for more nuanced handling of OOV words in certain contexts.

c. Can the word bigram model be implemented without any kind of smoothing? If not, try add-one smoothing. Is this kind of smoothing appropriate or do you need better algorithms? Why (not)?
Implementing the word bigram model without any form of smoothing is not viable, especially when dealing with natural language data where the vocabulary size can be extensive, and sparse data is common. Without smoothing, the model would encounter zero probabilities for unseen word bigrams, leading to unreliable predictions. Add-one smoothing, also known as Laplace smoothing, is attempted as a straightforward solution to handle unseen word combinations. While add-one smoothing ensures that no bigram has a zero probability, it may not be the most optimal choice for all scenarios. More advanced smoothing techniques such as Good-Turing smoothing or Kneser-Ney smoothing could provide better performance in certain situations by taking into account the frequency and distribution of word occurrences.

3. **Question Set #3**
   a. Same as Question #2, but replace the add-one smoothing with Good-Turing smoothing. What do you do when the number of words seen once are unreliable? What strategy do you use to smooth unseen words?
   Good-Turing smoothing is used as an alternative to add-one smoothing in the word bigram model. Good-Turing smoothing addresses the issue of unreliable estimations for words seen only once in the training data by adjusting the probability estimates based on the observed frequencies of rare events. When the number of words seen once is unreliable, Good-Turing smoothing adjusts the probabilities by redistributing the mass of rare events to unseen words, improving the model's generalization and handling of unseen word combinations.

Which of the language models at Question Sets #1, #2, and #3 is the best?
The word bigram model with Good-Turing Smoothing is identified as the most effective among the three language models. While letter bigram models capture fine-grained patterns at the letter level, word models with smoothing techniques provide better generalization and handling of unseen combinations at the word level. Good-Turing smoothing offers advantages over simple add-one smoothing by adjusting probabilities based on the observed frequencies of rare events, leading to more accurate and reliable predictions. However, the choice of the best language model depends on the specific requirements of the task and the characteristics of the training data, with considerations for vocabulary size, data sparsity, and the complexity of language patterns.