1. Define an N-gram and provide examples for unigram, bigram, and trigram using the sentence: "I love natural language processing."
    a. An N-gram is a contiguous sequence of N words from a given text. They are used for language modeling, text generation, and machine translation
        i. Unigram(N=1): [“I”, “love”, “natural”, “language”, “processing”]
        ii. Bigram(N=2): [(“I”, “love”), (“love”, “natural”), (“natural”, “language”), (“language”, “processing”)]
        iii. Trigram(N=3): [(“I”, “love”, “natural”), (“love”, “natural”, “language”), (“natural”, “language”, “processing”)]
    b. BOS is a special token indicating the start of a sentence. EOS is a special token indicating the end of a sentence. UNK is a placeholder token used for words not present in the vocabulary. In the example, “NLP” isn’t in the vocabulary.
        i. Example (“I love NLP”): [“<BOS>”, “I”, “love”, “<UNK>”, “<EOS>”]
2. Probability and Smoothing
    a. P(“sat”|”cat”) = Count(“cat sat”) / Count(“cat”) = 1/1 = 1
    b. Smoothing is necessary in N-gram models to handle zero probabilities for unseen word sequences, this prevents the model from assigning zero probability to valid sentences. Laplace smoothing adds 1 to the count of each N-gram and adjusts the denominator to account for the added counts, this is to ensure no probability is ever zero.
    c. P(“sat”|”cat”) = (1+1)/(1+7) = 2/8 = 0.25
3. Intrinsic and Extrinsic Evaluation
    a. Intrinsic evaluation measures the quality of the N-gram model directly, using metrics like perplexity. It focuses on how well the model predicts the next word. Extrinsic evaluation focuses on the model’s effectiveness in real-world tasks such as speech recognition or machine translation.
    b. Perplexity measures how well a language model predicts a given text set. It is useful because lower perplexity indicates better predictions, and it helps compare different language models.
    c. $PP = P(W)^{-1/N} = 0.01^{-1/10} = 1.58$