Language Models:
In this exercise you will explore different methods to estimate the probability of seen and yet unseen words and phrases. You will write a computer program that uses a training set to learn different language models and compares them according to how well they are able to predict the test set.

Before we continue we define how to measure how well a language model is. In information theory, perplexity is a measurement of how well a probability distribution or probability model predicts a sample. It may be used to compare probability models. A low perplexity indicates the probability distribution is good at predicting the sample. The perplexity of a test corpus $W$ is given by:

$$\text{Perplexity}(W) = 2^{H(W)} = \frac{1}{P(w_1w_2\ldots w_N)^{\frac{1}{N}}} = \sqrt[N]{\frac{1}{P(w_1w_2\ldots w_N)}} = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1\ldots w_{i-1})}}$$

Comments:
- All N-gram probabilities should be estimated using maximum likelihood estimation (MLE) as learned in class, that is, using simple counting.
- This exercise is not about text processing, we just use words as representing events, so ‘Trump’ is a different event from ‘trump’ and ‘trumps’. A word is series of characters between two spaces, and the word case sensitive.
- Remember to log scale your calculation as taught in class to prevent underflow.
- To debug your code you should always make sure that $\sum P(w_n) = 1$

Download these three books from Project Gutenberg and save them with the following names:
- alice.txt: Alice’s Adventures in Wonderland
- lookingglass.txt: Through the Looking-Glass
- sherlock.txt: The Adventures of Sherlock Holmes
These will be your training and testing data.

You should only use the text from the actual books not the text before and after the books and not the chapter headers either. You should ignore text until the first line that begins with ‘CHAPTER’ all lines that begin with ‘CHAPTER’ and everything after the line that begins in ‘THE END’.

Part 1:
The goal of this part of the assignment is to build a language model based on trigrams with smoothing for unseen trigram phrases but without smoothing for unseen unigrams. As in we assume that there are no words in our test set that were never seen in training.
In this part of the assignment you will need to train a trigram model with linear interpolation of the bigrams and unigrams as defined below.

\[
\hat{P}(w_n|w_{n-2}w_{n-1}) = \lambda_1 P(w_n|w_{n-2}w_{n-1}) \\
+ \lambda_2 P(w_n|w_{n-1}) \\
+ \lambda_3 P(w_n)
\]

such that the \(\lambda\)s sum to 1:

\[
\sum_i \lambda_i = 1
\]

You should decide what lambdas to use based on what gives you a lower perplexity on the test set. You should pick \(\lambda\)s intuitively. The specific values are less important - more importantly are the relations between them (e.g., you can pick \(\lambda=(0.4,0.1,0.5)\) since you think that the relation is \(\lambda_3 > \lambda_1 > \lambda_2\)).

Part 2:
Commonly after training a language model there will be words unseen during training or out of vocabulary words (OOV) most commonly nouns such as names. The goal of this part is to add probabilities for unseen words. For this you need to replace the first occurrence of each word in the training set with the pseudo word <UNK>. And the proceed to train the language model as before treating <UNK> as a regular word.

For training and development sets you should use the books “alice” and “lookingglass”. Divide them however you see fit for example use the first 90% of each book for train and the remaining 10% for development. For the test set and measuring the perplexity at the end use the book “sherlock”.
What must be handed in is:

1. The code that trains the two language models.
2. Your 3 lambda selections and an explanation as to why you chose those values.
3. The perplexity of the book “sherlock” using each model.

The assignment should be submitted via the submit system. Good luck!