First-Year Project 3: Natural Language Processing
Exercise 2: Statistics of natural language
Christian Hardmeier

The goal of this exercise is to get a feeling for the statistical properties of natural language in different corpora. You don't need to hand in a report for this exercise, but part of it can serve as the basis of Section 2 in your project report.

For this exercise, you will use the *training* sets of the two TweetEval tasks your group has selected. For comparison, you will also use a text of your choice from the downloaded from the Project Gutenberg repository. You will then calculate some basic statistics and create frequency lists of words and n-grams from these corpora.

Start by selecting an interesting comparison text from Project Gutenberg:
`https://gutenberg.org/browse/scores/top/`
Download the book in *Plain Text UTF-8* format. At the beginning and the end of the Gutenberg files, there are headers and footers with some metadata. These parts aren't interesting, so it's best to delete them manually by removing everything up to the line **START OF THE PROJECT GUTENBERG E-BOOK ...** and from the line **END OF THE PROJECT GUTENBERG E-BOOK ...** to the end of the file.

You now have three corpus files, two TweetEval corpora and one Gutenberg text. Tokenise them using the tokeniser you developed in the last exercise or some other suitable method. Then do the following analyses on each of the files:

1. Calculate the number of *types* and *tokens* and the *type/token ratio* of each text.
2. Create frequency word lists. Which are the 20 most common words in each text? Are there any meaningful similarities or differences between the most frequent words for each text?
3. Test your word distributions against Zipf's law by creating log-log plots of word rank against frequency.
4. Compare the most frequent words (excluding punctuation) with a general word list for English, such as the ones here:
   `https://www.wordfrequency.info/samples.asp`
   From this site, I suggest using the 5,000 word sample of the "top 220,000 word forms" list.
5. Create frequency lists of n-grams. If `text` is a Python list of tokens, you can use `list(zip(text, text[1:]))` to obtain a list of bigrams. What are the 20 most frequent bigrams and trigrams in your corpora, and what are their frequencies? Also experiment with higher-order n-grams. Can you see a reason why we rarely use n-gram orders higher than 3 in small corpora?