Build from scratch a spelling corrector in Python. It should include:

1. tokenization
2. Levenshtein distance-based non-word spelling correction
3. de-tokenization

As an example use case, consider a version of Jane Austen's *Sense and Sensibility* (available via nltk's gutenberg corpus) corrupted by random insertions, deletions, and substitutions. See for reference `gen_corrupted.py`.

Your spelling correction function:

- should accept a document as a single string and return the corrected document as a single string.
- may use *only* standard libraries and `numpy`.
- may use this English word list, which is ordered by decreasing frequency.

Document some specific cases in which your spelling corrector worked well and in which it worked poorly. Why is it working poorly and how could it be improved?

You may work in a group of 1 or 2. Submissions will be graded without regard for the group size. You should turn in a document (`.txt`, `.md`, or `.pdf`) answering all of the red items above. You should also turn in Python scripts (`.py`) for *each* of the blue items.

Spelling correction is not an easy problem and you will not solve it. Make some defensible assumptions, state them, and get a "minimum viable" pipeline working. If you have time afterward to go back and handle some edge cases, feel free.