Word Abbreviator

High Level View

1. **Load** file as a string.
2. **Convert** each string into the correct format.
3. **Find** all abbreviations.
4. **Score** all abbreviations.
5. **Remove** duplicate abbreviations.
6. **Find** all min scoring abbreviations
7. **Save** all min scoring abbreviations.

Pseudocode

1. **Load** file:
    - **Load** file.
2. **Convert** each string into the correct format:
    - For each word:
        - **Make** it uppercase.
        - **Remove** special characters.
        - **Split** string at **special** characters and **empty** spaces.
3. **Find** all abbreviations:
    - For each word:
        - **Find** all possible **sequential** (each some n after, first is always the first character in string) 3 letter words.
        - **Determine position** of each character in the abbreviation.
        - **Store** as a list of abbreviations.
4. **Remove** duplicate abbreviations.
    - **Find** all duplicate abbreviations across words.
    - For each abbreviator list:
        - **Remove** abbreviations **based** on duplicate abbreviations list.
5. **Score** abbreviations:
    - For each abbreviations list:
        - **Score** all abbreviations.
6. **Find** all min scoring abbreviations:
    - For each abbreviations list:
        - **Find** min score.
        - Save all abbreviations with taht score in a list
7. **Save** abbreviations:
    - For each min scoring abbreviation list:
        - Save to file

Notable Decisions

Testing

Used **unit tests** to test **most important** parts.

- Word splitting
- Finding abbreviations
- Determining position for each abbreviation character
- Removing duplicate abbreviations
- Scoring abbreviations
- Finding mind abbreviations

Code Structure

- Due to the use of unit tests, I had to separate logic between multiple functions
  - find abbreviations, find all abbreviations
  - find min scoring abbreviations, find all min scoring abbreviations
- Separated some functions into a utility script for better separation and readability.

Determine Position

To **determine** the position of each character in the abbreviation based on the word it comes from.

When finding all abbreviations, I **combined** the split words into a single string, with each word being separated by #.

```
split_string = ['HELLO', 'WORLD']
combined_string = 'HELLO#WORD'
```

To determine position of the character **relative** to character is comes from, I used # as **marker**. Out of bounds was also treated as a marker.

- If # is 1 place before it, the position = first
- If # is 2 places before it, the position = second
- If # is 1 place after it, the position = last