function: wordseq_successor_counts

1. Inputs: filename, sequence size, punctuation (boolean), case-sensitive (boolean)
    a. For the booleans, true means punctuation is included/case is not ignored respectively, and false means the opposite
2. Output: map of words to their counts
3. The function 'word_no_longer_counts_file' is called, which returns a list of all words in the specified input file.
4. A counter is set up that keeps track of how many times a specific successor word appears in the file.
5. To count the first sequence and successor, the current sequence is obtained by choosing a portion of the list of words from the beginning until the specified sequence size index, but not including that index.
6. The successor is set as the value of wordlist at the index 'sequence size'
7. To count the remainder of the sequences and successors, for a specific word in the remainder of the word list (from index sequence size to the end) the current sequence is reset to the next sequence by calling the function 'next_seq' and specifying the next sequence.
8. Afterwards, the succeeding sequence is set to equal the word from the list.
9. The counter is incremented for the word and returned at the end (after looping through every word). Will include punctuation if include_punc is True.
10. Will ignore case if is_case_sensitive is False.

function: wordseq_successor_frequencies

1. Inputs: filename, sequence size, punctuation (boolean), case-sensitive (boolean)
    a. For the booleans, true means punctuation is included/case is not ignored respectively, and false means the opposite
2. Output: map of words to their frequencies
3. The previous function, described above, is called.
4. To determine percentages, for every element in the map, the count is divided by the sum of the total counts of all elements.
5. The result is mapped to the element and the map is returned.