1. **Word Frequency Counter:**
   - **Task:** Write a Java program that reads a text file (e.g., "shakespeare.txt") and counts the frequency of each word. Ignore case and remove punctuation from words.
   - **Requirements:**
     - Use a `TreeMap<String, Integer>` to store word frequencies.
     - Read the file line by line, split each line into words, and update the word counts.
     - Print the word frequencies at the end.
2. **Reverse Mapping:**
   - **Task:** Implement a method that reverses a given map (keys become values, and values become sets of keys).
   - **Requirements:**
     - The method should take a `Map<K, V>` as input and return a `TreeMap<V, Set<K>>`.
     - For each entry in the input map, add the key to the set associated with its value in the output map.
     - Ensure that the output map is sorted by values.
3. **Find Most Frequent Set:**
   - **Task:** Write a method that finds the most frequent set of words of a given size in a `TreeMap`.
   - **Requirements:**
     - The method should take a `TreeMap<K, Set<V>>` and an integer `n` as input.
     - Iterate through the keys in descending order and find the first set with size `n`.
     - Return a map containing the key and its associated set.
4. **Find All Sets of a Given Size:**
   - **Task:** Implement a method that finds all sets of words of a given size in a `TreeMap`.
   - **Requirements:**
     - The method should take a `TreeMap<K, Set<V>>` and an integer `n` as input.
     - Iterate through the keys in descending order and collect all sets with size `n`.
     - Return a map containing the keys and their associated sets.
5. **Find Kth Largest Set:**
   - **Task:** Write a method that finds the kth largest set of words of a given size in a `TreeMap`.
   - **Requirements:**
     - The method should take a `TreeMap<K, Set<V>>`, an integer `n`, and an integer `k` as input.
     - Iterate through the keys in descending order and find the kth set with size `n`.
     - Return a map containing the key and its associated set.