My program uses a number of structures to generate the desired output including a Trie structure to store the dictionary words and multiple character arrays to temporarily store the words that we need to work with.

The Trie structure is defined by our struct node class in my header file where we assign a number of different fields to the node class. These fields represent the character I wish the node to hold, the children of the Node, as well as an isWord, occurrence, prefix, and numChild variables. These fields are described below.

isWord - used to determine whether the character in the Node we are currently at completes a full word in our Trie
occurence - used to keep track of the number of times this word appears UNIQUELY in my data file
prefix - used to keep track of the number of times this word appears as a PREFIX in my data file
numChild - used to keep track of the number of children nodes that any given node has

The child Node field of our Node class is initiated as an array of pointers of size 26 for the 26 letters in the English alphabet. This means that for every child we create, child will point to a pointer array of size 26. I initiate each of these 26 addresses to NULL and activate each address as we need them.

The benefit to using an array of pointers is that we can use a CHAR_TO_INDEX calculation to assign children and access into the Trie runs in O(n). At any point, the worst case to find a word in our Trie will be n where n is the length of the word.

The way I dynamically allocate memory for the array that holds the dictionary and data words before they are used saves us significant amounts of memory. I was pretty diligent in freeing all addresse

s that were no longer required as well. In this r
espect, if we are trying to store k words of m len
gth in my Trie, the Trie will take up memory of si
ze O(k*m) which is ideal.

Finally, my program outputs the CORRECT output in
EVERY test case except for 6 and 8 where only two
fields are incorrect. The program runs within the
 allotted time frame for each test case and produc
es the correctly named output files. For some rea
son, autograder does not like me and keeps giving
me a 0.