I decided to use a linked list to create my code, much like what was described on Wednesday’s lecture. I made a linked list with a size completely dependent on the inputs (so if the input was the entire English vocabulary, it would be quite large). The linked list points to multiple other nodes depending on the word inputs, so the path can be fairly complex.

I created four classes: Trie, Autocomplete, Main, and AutocompleteTests. Main doesn’t contain anything, and AutocompleteTests contain only tests for the main code. Trie is similar to the TreeNode classes we’ve used in the quarter, creating and keeping track of various nodes in the linked list. It includes a toString method, getChild method, and getWords, which is to be done recursively by backtracking. Autocomplete contains the rest of the important methods, including place (an insert method), find (a search method), and finish, which returns the list created using the Trie class.

The time complexity for the code should be O(n), where n is the longest string in the trie. This ends up being fairly quick with the limited inputs given, but unfortunately wouldn’t be able to handle very large libraries of words. I wish I could have made it faster, but this was unfortunately the best I could do in the time I gave myself (thanks procrastination).

***Example Test Method***

@Test  
public void oneChar() {  
 Autocomplete z = new Autocomplete();  
 z.place("hello");  
 z.place("high");  
 z.place("seattle");  
 z.place("seatac");  
 z.place("see");  
 z.place("hollow");  
 z.place("how");  
 z.place("hell");  
  
  
 List input = z.finish("h");  
 List expected = new ArrayList();  
 expected.add("hell");  
 expected.add("hello");  
 expected.add("high");  
 expected.add("hollow");  
 expected.add("how");  
  
 *assertEquals*(expected, input);  
}

Z is the name of the Autocomplete linked list, and the words: hello, high, seattle, seatac, see, hollow, how, and hell were added. When searching with one character *h*, the resulting list is given as [hell, hello, high, hollow, how], because all of the words began with the letter h.