Lab 4 (Due March 8th)

Instead of using a linked list to resolve collisions, as in separate chaining, use a binary search tree. That is, create a hash table that is an array of trees. You can use the hashChain.java program as a starting point and the Tree class from the tree.java program. To display a small tree-based hash table, you could use an in-order traversal of each tree.

The advantage of a tree over a linked list is that it can be searched in O(logN) instead of O(N) time. This time savings can be a significant advantage if very high load factors are encountered. Checking 15 items takes a maximum of 15 comparisons in a list but only 4 in a tree.

Duplicates can present problems in both trees and hash tables, so add some code that prevents a duplicate key from being inserted in the hash table. (Beware: The find() method in Tree assumes a non-empty tree.) To shorten the listing for this program, you can forget about deletion, which for trees requires a lot of code.