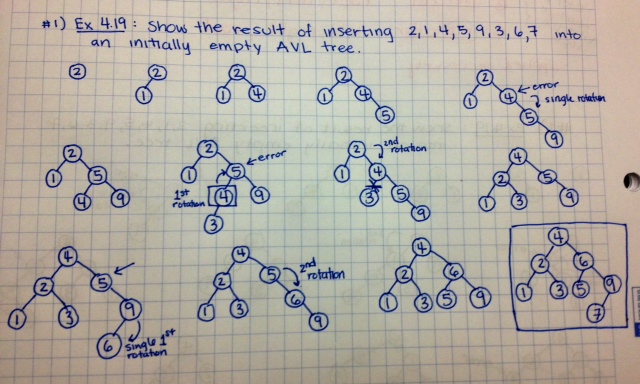
**COMS W3134 Fall 2014  
Homework 3  
Name: Akiko Iwamizu**

**UNI: ami2119**

**Written**

1. Weiss, Exercise 4.19: Show the result of inserting 2, 1, 4, 5, 9, 3, 6, 7 into an initially empty AVL tree.



1. Weiss, Exercise 4.31: Write efficient methods that take only a reference to the root of the binary tree, T, and compute:
   1. The number of nodes in T

private int numNodes(BinaryNode<AnyType> t)

{

if(t == null)

return 0;

else

return(numNodes(t.left) + numNodes(t.right) + 1);

}

* 1. The number of leaves in T

private int numLeaves(BinaryNode<AnyType> t)

{

if(t == null)

return 0;

else if(t.left == null && t.right == null)

return 1;

else

return(numLeaves(t.left) + numLeaves(t.right));

}

* 1. The number of full nodes in T

private int numFullNodes(BinaryNode<AnyType> t)

{

if(t == null)

return 0;

else if(t.left != null && t.right != null)

return(numFullNodes(t.left) + numFullNodes(t.right) + 1);

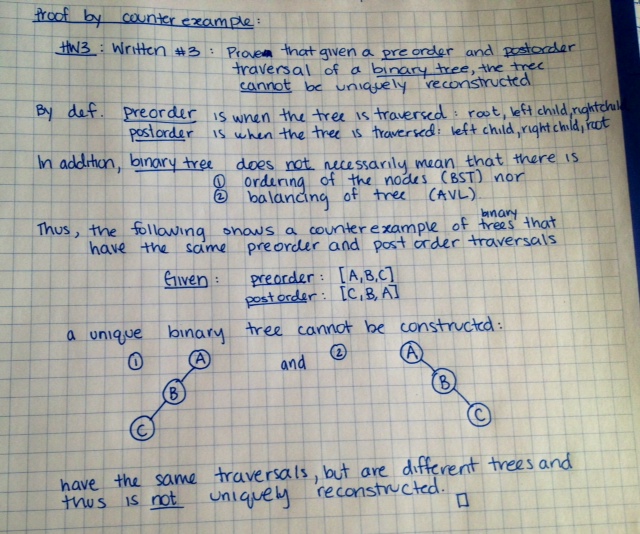
else

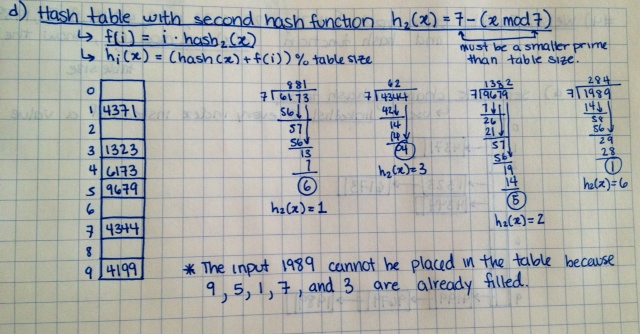
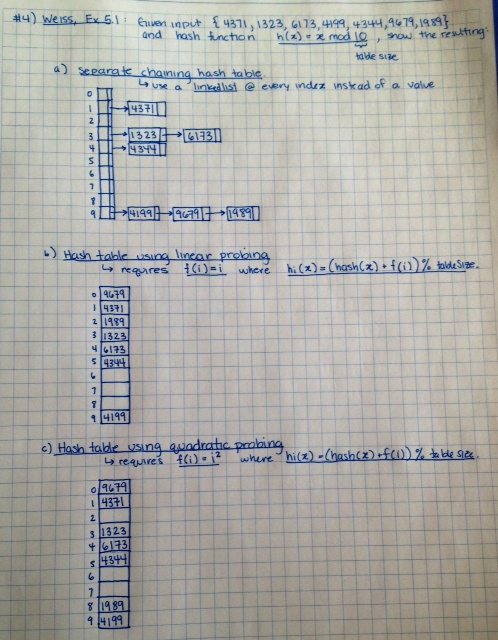
return(numFullNodes(t.left) + numFullNodes(t.right));

}

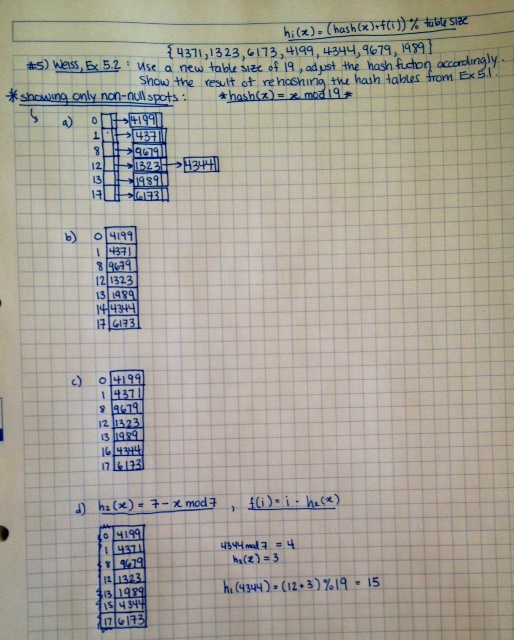
* 1. What is the running time of your routines?

The running time is the depth of the largest node, so for the average case is O(logN) but the worst case scenario is **O(N) for all of the algorithms above.**

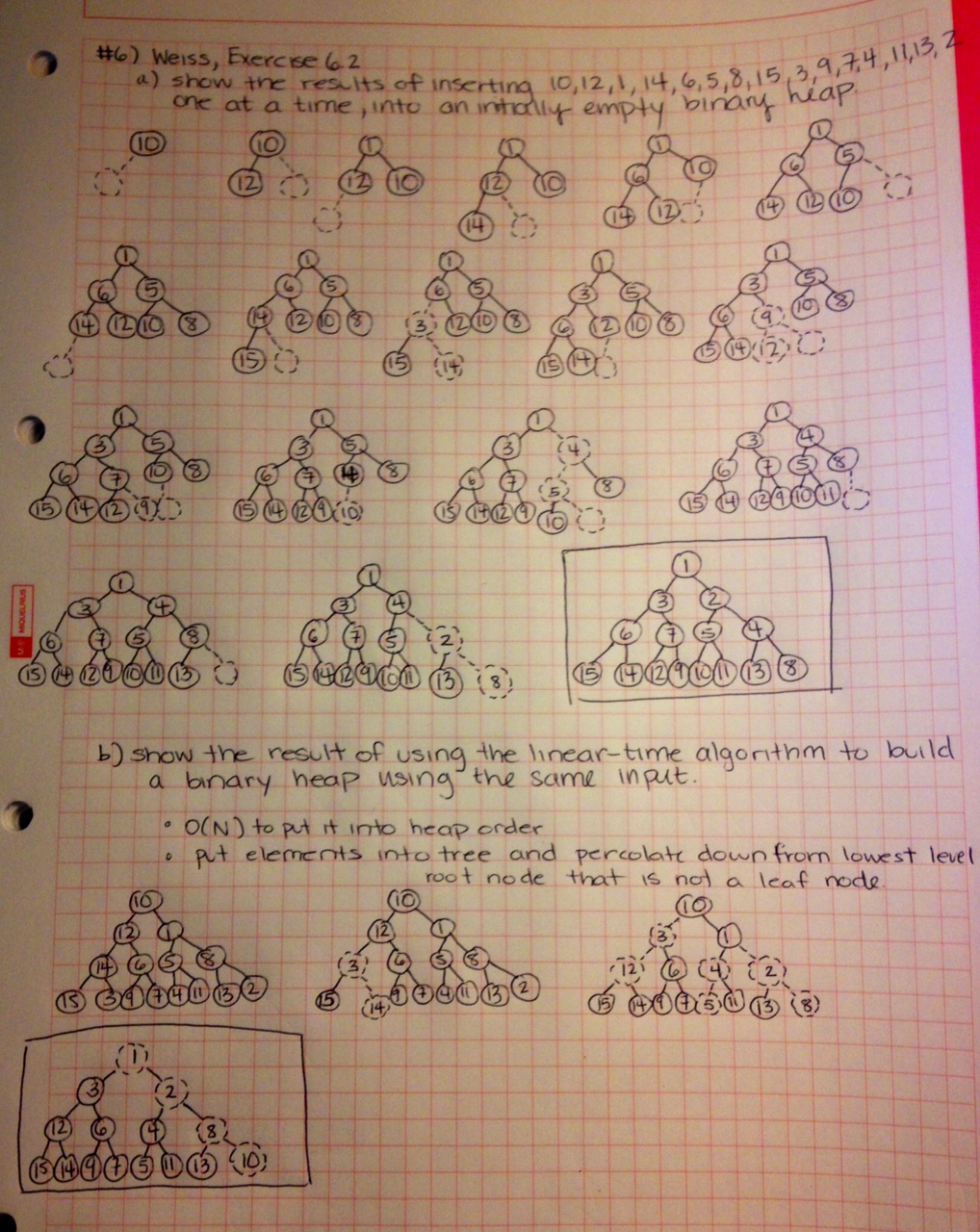
1. Prove that given a preorder and a postorder traversal of a binary tree, the tree **cannot** be uniquely reconstructed.  
   
2. Weiss, Exercise 5.1: Given input {4371, 1323, 6173, 4199, 4344, 9679, 1989} and a hash function h(x) = x mod 10, show the resulting:



1. Weiss, Exercise 5.2 - Use a new table size of 19, adjust the hash function accordingly. Show the result of rehashing the hash tables in Exercise 5.1.



1. Weiss, Exercise 6.2 (show the contents of the heap after each step)



1. Weiss, Exercise 6.8: Show the following regarding the maximum item in the heap:

