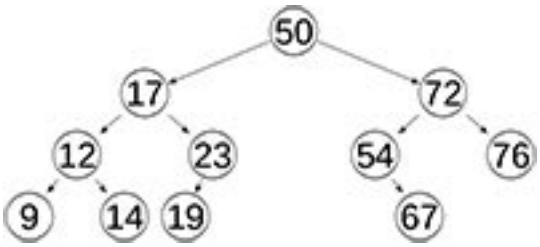
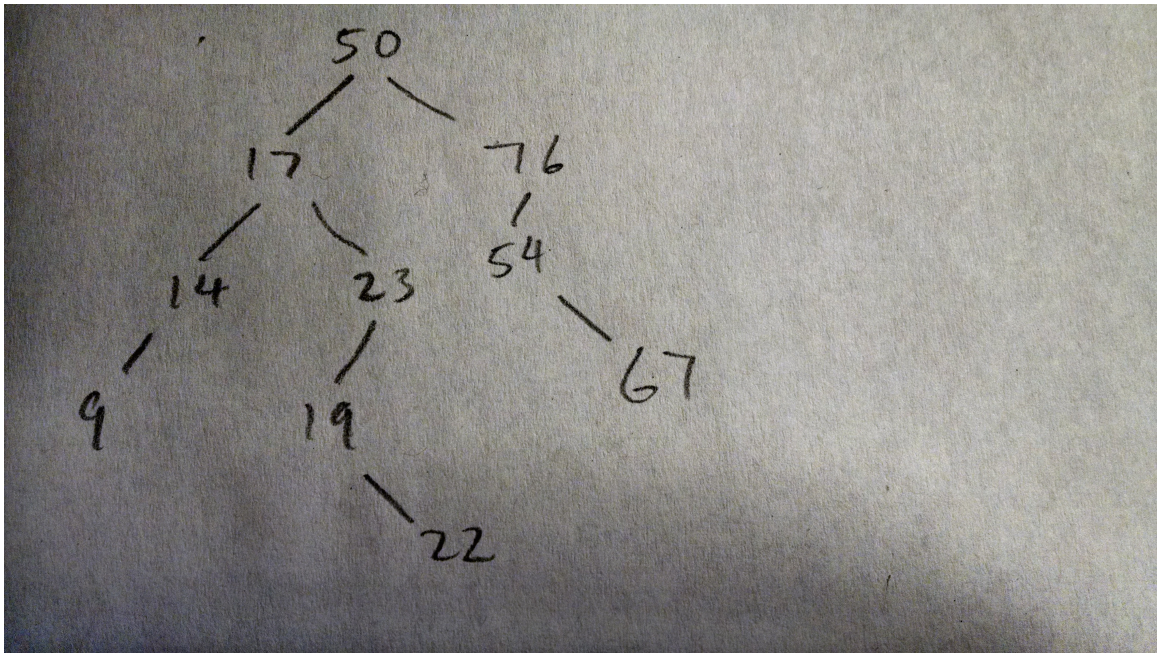
CS61B Spring 2016 Secret Section 3 Worksheet

Week of March 14, 2016

1. Warm up: Given a diagram of a BST, practice inserting and deleting nodes. (There may be more than one way to go about each of these.)

   - Insert 22.
   - Delete 12 and 72.
   - How should you delete 50?





   This is the solution assuming we promote the smallest descendant of the deleted node's right subtree. We can just as well promote the largest descendant of the deleted node's left subtree.

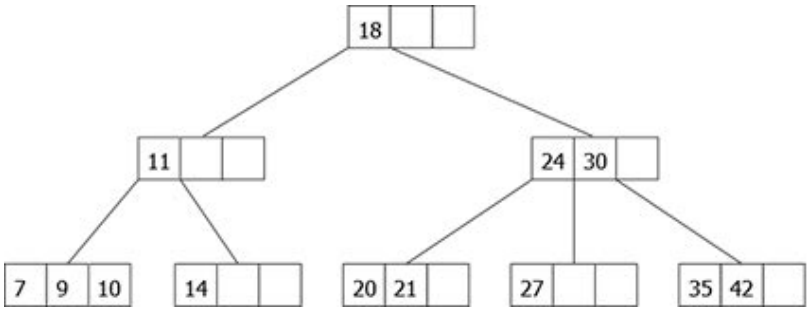
   We would delete 50 by promoting either 54 or 23 as the root, and re-attaching their children accordingly.

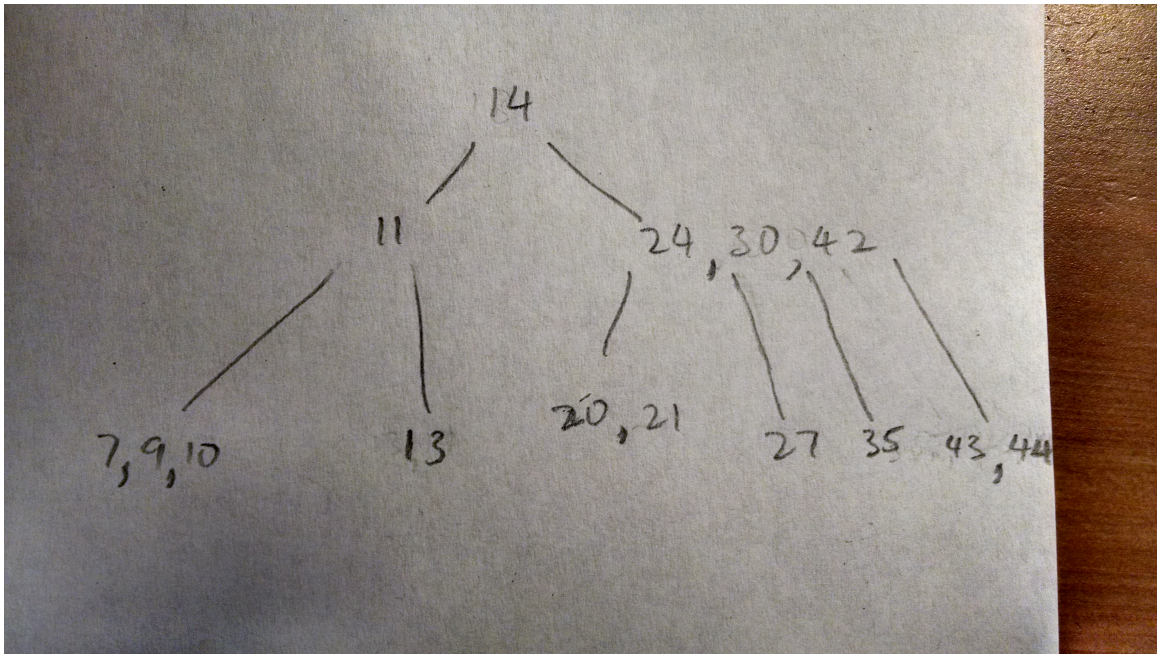
2. Implement isBalanced. A balanced tree assures that its left and right subtrees differ by no more than 1. The getHeight method will come in handy.

```java
public static int getHeight(TreeNode root) {
 if (root == null) return 0; // Base case
   return Math.max(getHeight(root.left),
   getHeight(root.right)) + 1;
}

public static boolean isBalanced(TreeNode root) {
   if (root == null) return true;

   int heightDiff = getHeight(root.left) - getHeight(root.right);
   if (Math.abs(heightDiff) > 1) {
       return false;
   } else {
       return isBalanced(root.left) && isBalanced(root.right);
   }









}
```

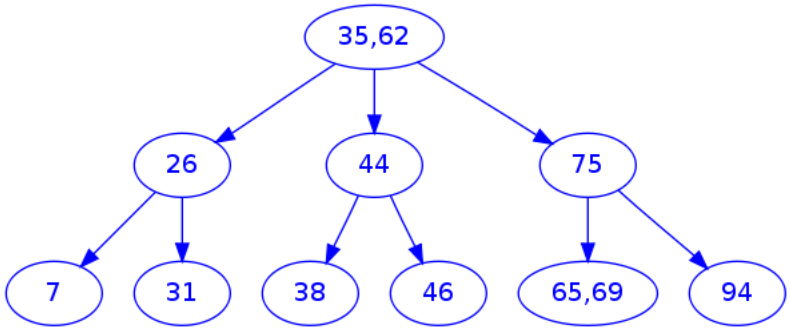
3. Insert and delete some elements from the following 2-3-4 tree.

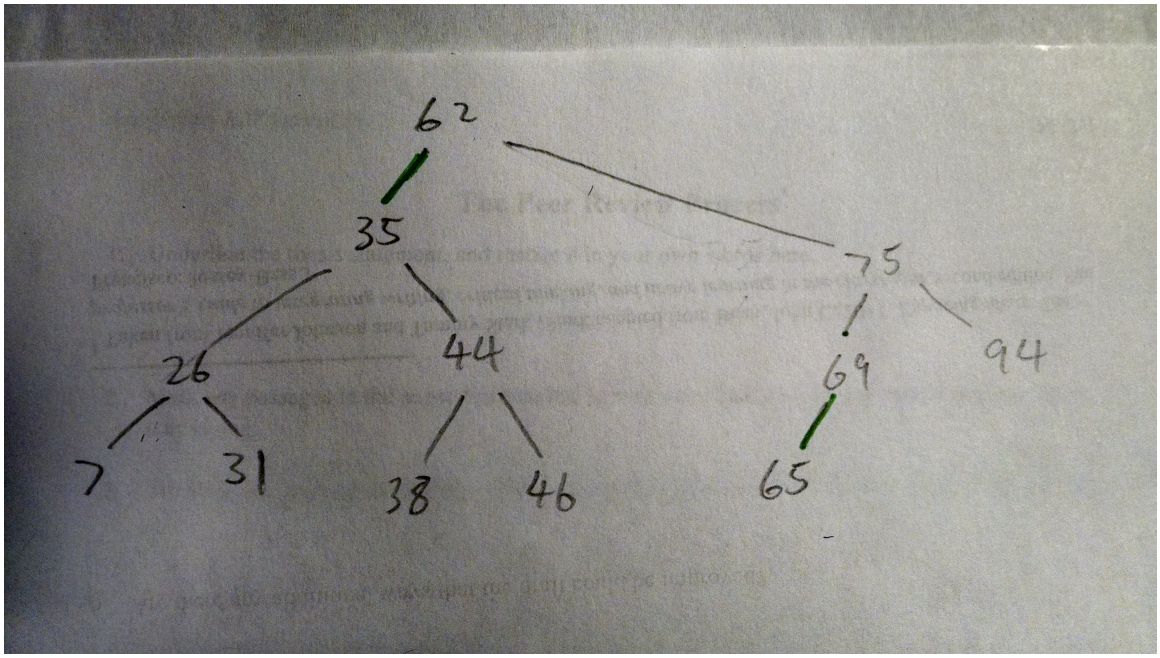
   - Insert 13
   - Delete 18
   - Insert 43
   - Insert 44





4. Convert the following 2-3 tree into a left-leaning red-black tree.





(Red edges are shown as green.)