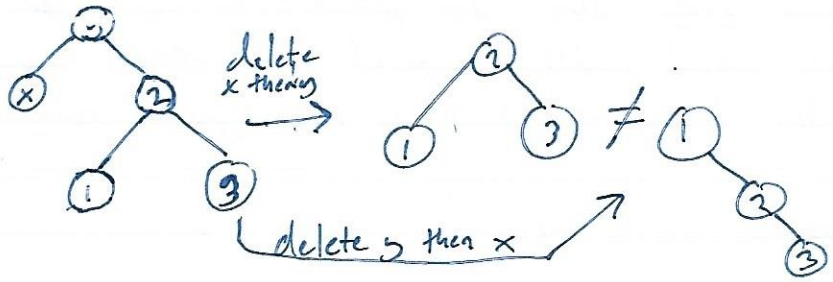
HW 3 - CS430

Chris Morcom

12.3-4

Deleting elements from a BST is NOT commutative.



13.1-6

Given that a Red-Black tree can have a height of $2n+1$, a complete, and thus maximized tree has $2^{2n+1}-1$ nodes, where nodes alternate red and black. A complete tree with all black nodes is minimized with $2^{n+1}-1$ nodes

13.3-4

The function can only modify a red child of a node. Because the root and parent of a root is automatically black (due to line 16), the loop only runs when the tree has a depth of 2 or more. When z is at depth 2, the parent or the root could change to red, but due to line 16, the root will not change to red. And due to the initial condition, which means that the root and parent are black, the var T.nil.color cannot change to red.

13.4-6

When the function finds the left child of the root (line 2), w is assigned to its right sibling. Line 4 checks the color of w, which, if w is red, then the parent is not red. In any other case, the child nodes, x and w would have to be red, which violates the property that no two adjacent nodes can be red.

Hence, the professors are not right.

14.2-2

In the worst case, the delete operation on a root of a RB-Tree causes $O(n)$ updates to the Tree's nodes. This contradicts the $O(n \log n)$ performance outlined in both the notes and in Theorem 14.1 (p 346-347) in the textbook.