
CSCE 2211 Exercises
Exercises (4): Self-Balancing Trees

---

1. Find the minimum number of nodes required to construct an AVL Tree of height h = 5.

   Solution:
   N(h) = N(h-1) + N(h-2) +1   for h > 2 with N(1) = 1 and N(2) = 2
   N(3) = 4, N(4) = 7 so that N(5) = 12

---

2. Find the maximum height required to construct AVL Tree of 375 nodes.

   Solution:
   Using the formula h ≤ floor (1.44 log n +0.5), then for n = 375, h (max) = 12

---

3. Find the maximum height required to construct AVL Trees of 1K nodes and 1M nodes.

---

4. Suppose the number of nodes in an AVL tree is ***(n 2^n).*** Which complexity is **True** from the following worst case search costs for an element in that tree:
   (a) ***O(n log n)***   (b) ***O(n 2^n)***   (c) ***O(n)***   (d) ***O(log n)***

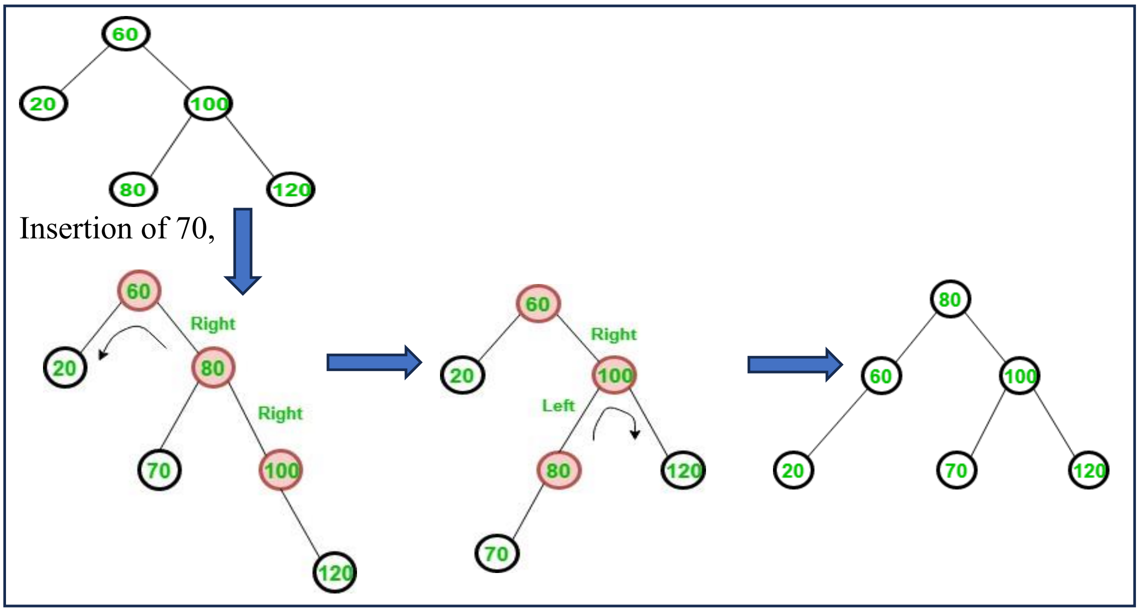
   Explain the reasoning behind your choice.

   Solution:
   Choice (c) is True. Complexity is *O(n).*
   Why?: The worst case cost of search in an AVL tree is ***O(log N)***, where *N* is the total number of nodes. Since *N* = ***n 2^n, then log N = log n + n log 2 = log n + n.***
   O(log N) = O(log n) + O(n) = O(n)

---

5. Consider the sequence of keys: 60, 100, 20, 80, 120, 70. Show the steps of insertion in an AVL tree of this sequence.

   Solution:
   After BST insertion of 60, 100, 20, 80, 120, the AVL tree is:



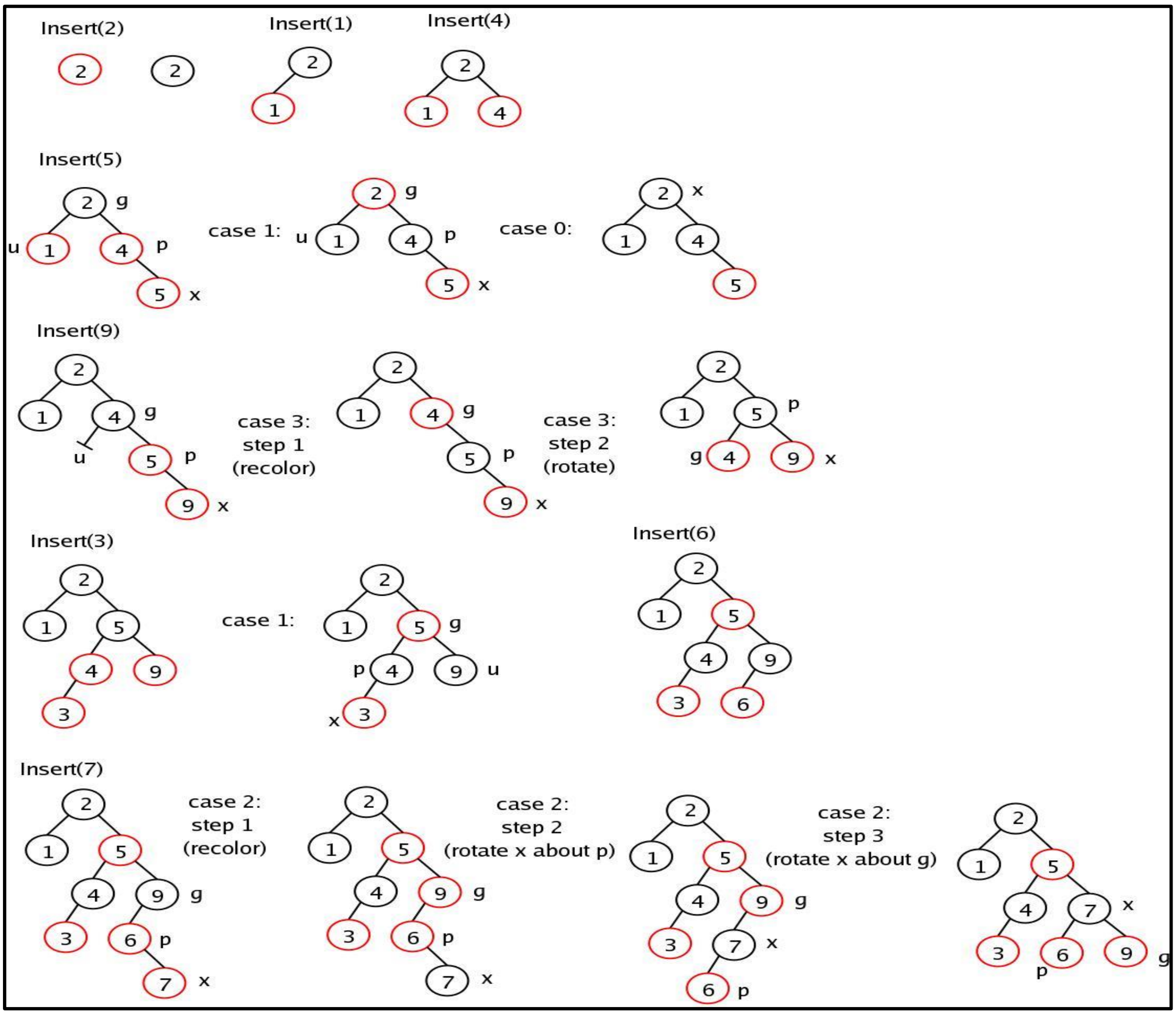
---

6. Insert the following sequence of keys into an AVL tree, starting with an empty tree (show each step): 12, 24,14, 27, 35, 17,19, 22.
   In the AVL tree you finally got, delete 27 (show the steps).

---

7. Insert the following sequence of keys into Red-Black tree, starting with an empty tree (show each step): 2, 1, 4, 5, 9, 3, 6, 7

   Solution:



---

8. Insert the following sequence of keys into Red-Black tree, starting with an empty tree (show each step): 21, 11, 35, 51, 60

---

9. Indicate for each of the following statements if it is true or false (Justify your answers).
   (a) The subtree of the root of a red-black tree is always itself a red-black tree.
   (b) The worst-case time complexity of the insert operation into an AVL tree is *O (log n)*, where n is the number of nodes in the tree.
   (c) The worst-case time complexity of the search operation into a Red-Black tree is *O (log n)*, where n is the number of nodes in the tree.

Solution:

(a) **FALSE.** The root of a Red-Black must be black, by definition. It is possible for the child of the root of a red-black tree to be red. Therefore, it is possible for the subtree of the root of a red-black tree to have a red root, meaning that it cannot be a red-black tree. So, the statement is false.

(b) **TRUE.** The work of all AVL tree operations is O(h) where h is the height of the tree. AVL rotations ensure that h is $O(\log n)$. Therefore, insertion must be $O(\log n)$.

(c) **TRUE.** For a Red-Black tree, the height is $h \le 2 \log (n+1)$, then $h = O(\log n)$. The queries SEARCH, MIN, MAX, SUCCESSOR, and PREDECESSOR all run in $O(h)$. Hence, all queries run in $O(\log n)$ time