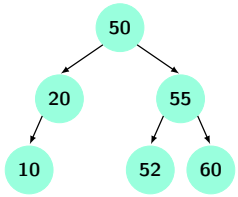
Week 12 Preparation

(Solutions)

Useful advice: The following solutions pertain to the preparation problems. You are strongly advised to attempt the problems thoroughly before looking at these solutions. Simply reading the solutions without thinking about the problems will rob you of the practice required to be able to solve complicated problems on your own. You will perform poorly on the exam if you simply attempt to memorise solutions to these problems. Thinking about a problem, even if you do not solve it will greatly increase your understanding of the underlying concepts.

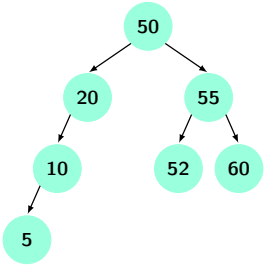
Problems

Problem 1. Insert 5 into the following AVL tree and show the rebalancing procedure step by step.

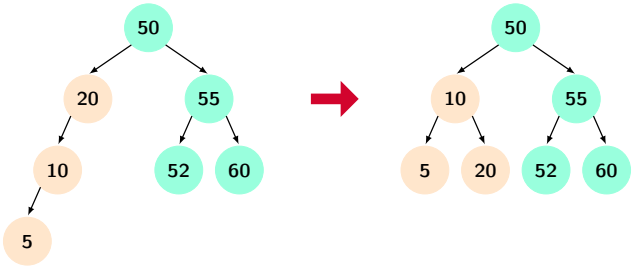


Solution

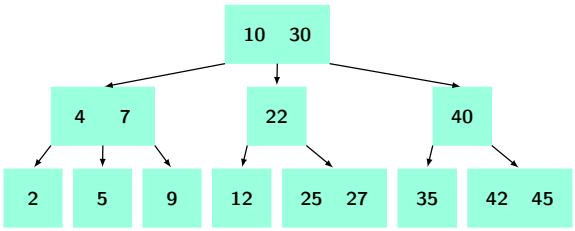
We insert 5 using the ordinary BST insertion procedure and we obtain:



This tree is imbalanced, as the node containing 20 has a height 2 left subtree, and a height 0 right subtree, resulting in a balance factor of 2. To rectify this, we rotate the node 10:



Problem 2. Sequentially perform the following operations on the given 2-3 Search Tree:



(a) Insert key 6

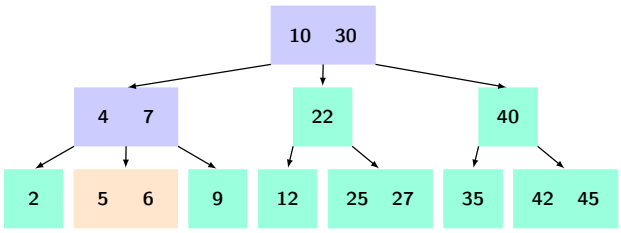
(b) Insert key 7

(c) Insert key 50

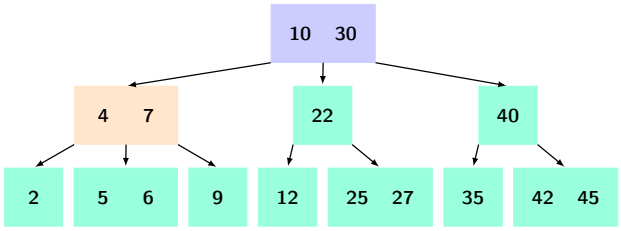
Solution

The purple nodes are the nodes traversed during the operation, and the orange is the final node in the search.

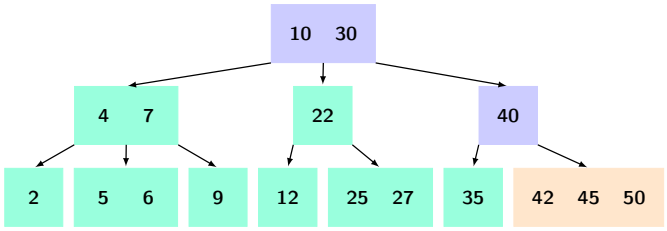
(a) Inserting 6 we traverse the tree arriving at the 2-Node containing 5. This now turns into a 3-node with 6 in the right slot.



(b) Inserting 7 we traverse the tree and we find the key 7 already exists, thus no change is made.

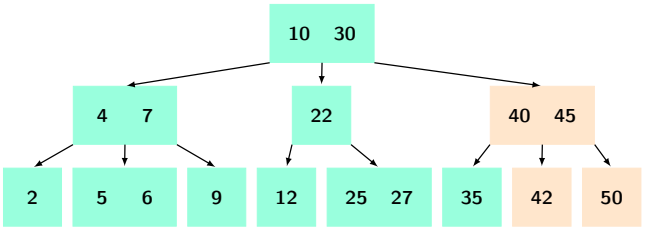


(c) Inserting 50 we traverse the tree and arrive at a 3-Node containing 42 and 45. We create a temporary 4-Node containg the 3-Nodes keys and 50.

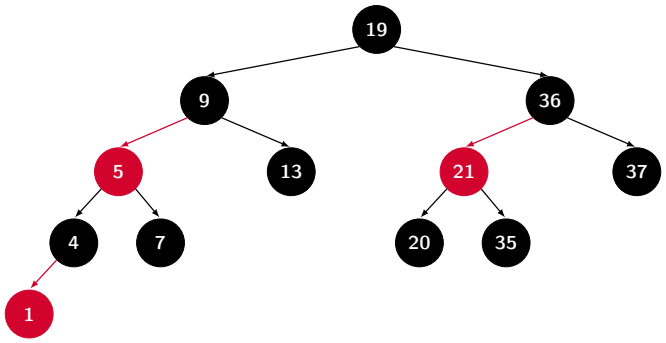


Since a 4-Node isn't a valid node in a 2-3 Search Tree we split the 4-Node into two 2-Nodes and promote the middle key 45 to the parent of the original 3-Node. Since all nodes in the path are

2-Nodes or 3-Nodes we have completed the insertion.



Problem 3. Consider the following Left-Leaning Red-Black Tree:



Visualise the tree after performing each of the following actions:

(a) Insert key 12

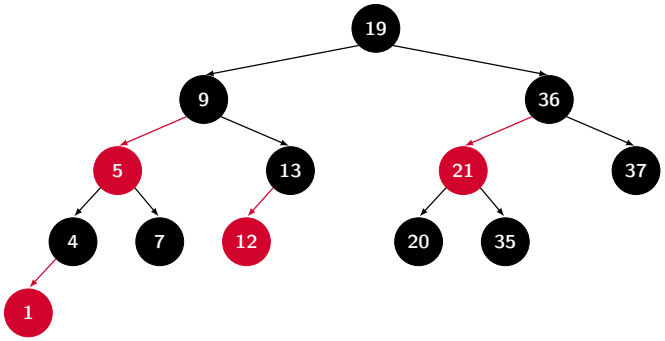
(b) Insert key 8

(c) Insert key 15

Assume the effects of previous insertions persist for the next insertion.

Solution

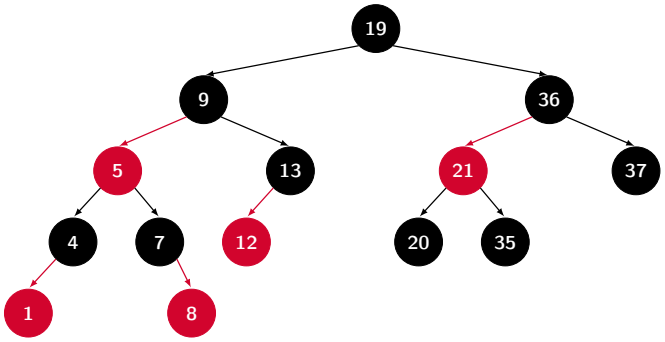
(a) To insert the key 12, we must first traverse the RBT to ensure it does not already exist, and only then we will insert it as a red node down the tree. In this case, key 12 would be inserted as the left child of 13.



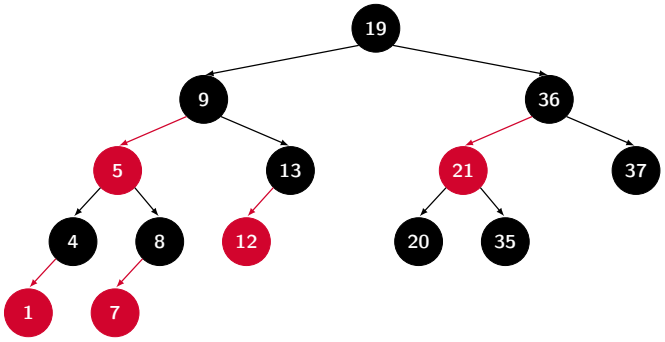
It is important to note that any key must always be inserted as a red node. After inserting 12, we

notice that it does not violate any of the required RBT properties and hence no further adjustments are needed.

(b) Key 8 would be inserted as the right child of 7 in the RBT.

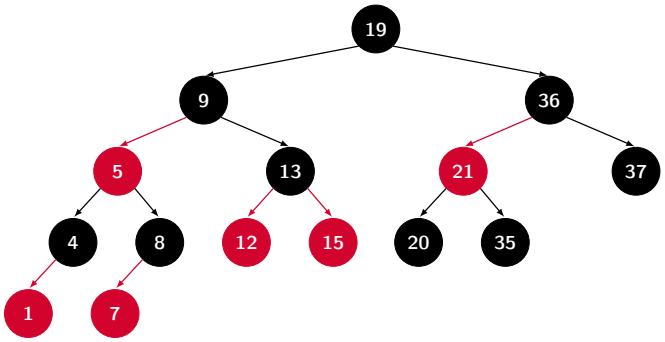


However, a red node cannot be the right child in a Left-Leaning RBT. Hence, we must perform an adjustment to fix this situation. In this case, a left rotation is required, which will effectively switch the positions of keys 7 and 8, where key 7 will become red.

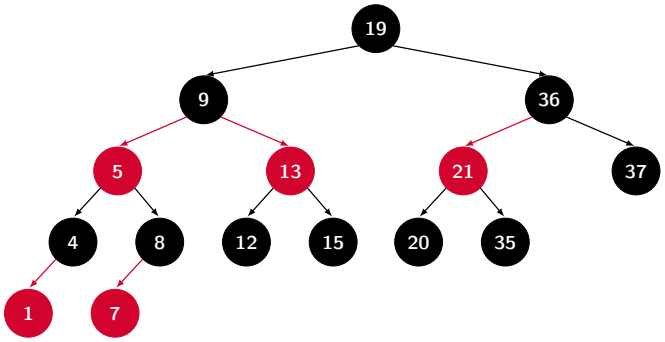


The properties of the RBT are now satisfied as the red key 7 is now a left child.

(c) Finally, inserting key 15 would place it as the right child of key 13.



As a result, key 13 has two red children. To fix this problem, we can flip the colours of key 13 and it's direct children, keys 12 and 15.



It is important to make sure we keep performing adjustments until everything is satisfied. The same issue, this time involving key 9 and it's direct children has appeared. To solve this, we can apply the same type of adjustment to obtain our final representation.