Department of Computer science and Engineering

PES UNIVERSITY

UE19CS202: Data Structures and its Applications (4-0-0-4-4)

AVL TREES

Abstract

Representation of AVL tree, operations performed on AVL, insertion, Deletion, Search

Dr.Sandesh and Saritha

Sandesh_bj@pes.edu

Saritha.k@pes.edu

Representation of AVL Trees

```
struct NODE
{
    int info;
    struct NODE *left,*right;
    int balancefactor;
};
```

AVL operations

The Different operations performed on AVL tree are

1. Insert
2. Delete
3. Search

Insertion in AVL tree:

For example construct AVL Tree for the following sequence of numbers-

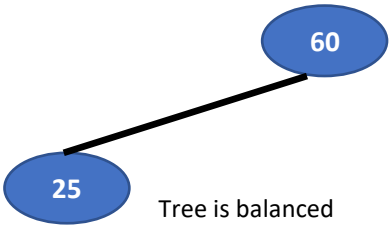
60, 25, 70, 10, 5

Solution-
Step-01: Insert 60

Tree is balanced

Step-02: Insert 25

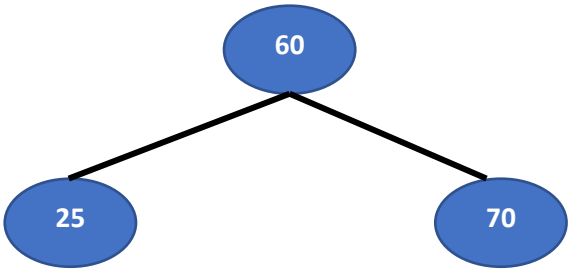
As 25 < 60, so insert 25 in 60's left sub tree.



Tree is balanced

Step-03: Insert 70

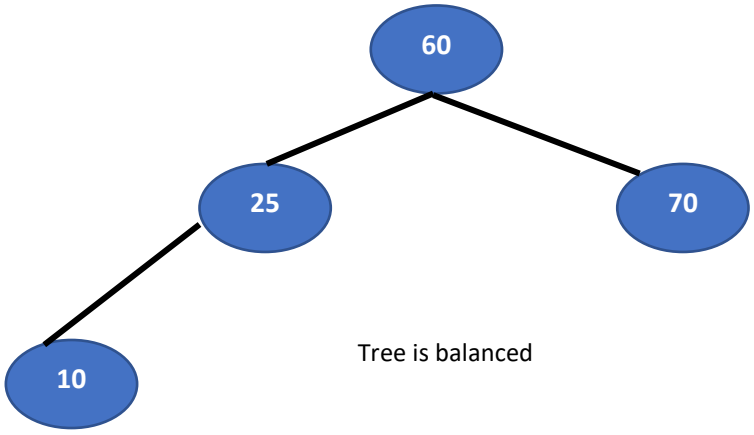
As 70 > 60, so insert 70 in 60's right sub tree.



Tree is balanced

Step-04: Insert 10

As 10 < 60, so insert 10 in 60's left sub tree.
As 10 < 25, so insert 10 in 25's left sub tree.



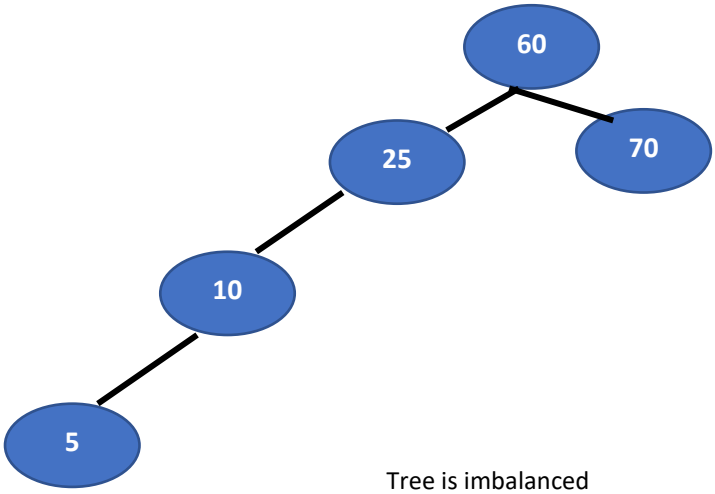
Tree is balanced

Step-05: Insert 5

As 5< 60, so insert 5 in 60's left sub tree.

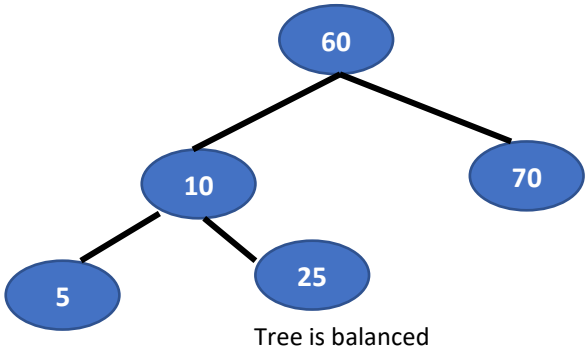
As 5 < 25, so insert 5 in 25's left sub tree.

As 5 < 10, so insert 5 in 10's left sub tree.



Tree is imbalanced

To balance the tree, find the first imbalanced node on the path from the newly inserted node (node 5) to the root node. The first imbalanced node is node 25. Now, count three nodes from node 25 in the direction of leaf node. Then, use AVL tree rotation to balance the tree. After applying RR Rotation



Tree is balanced

Algorithm for insertion:

Step1: insert an element into the tree using Binary search tree method.

Step2: check the balance factor after inserting an element.

Step3: if the balance factor of every node is less than or equal to one then proceed to next operation. Otherwise tree is imbalanced, perform the suitable rotation to make it balanced and then proceed to next operation.

2. Deletion operation

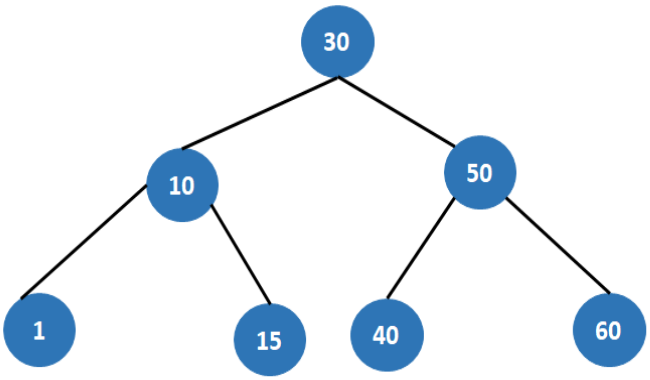
Deletion of a node from a tree decreases the height of the tree, which might lead to unbalanced tree.

The steps of deletion of a node in AVL are –

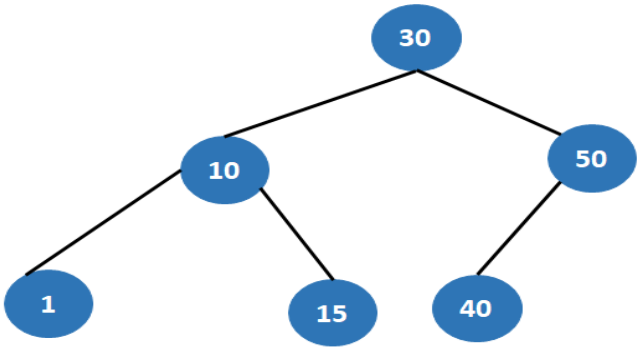
1. Binary Search Tree deletion algorithm is used to delete a given key from the AVL tree.
   2. Update the height of the current node of the AVL tree after the deletion.
   3. Calculate the Balance factor at the current node by finding the difference between height of left sub-tree-height of right sub-tree.
   3a. If the Balance factor > 1 which means the height of the left sub-tree is greater than the height of the right sub-tree. This indicates left-left or left-right case. If the Balance factor of left subtree is greater than 0 then that confirms left-left case, if it less than 0 then that confirms left-right case. If it is equal to 0, then this we can either consider this as a left-left case or as a left-right case. For left-left case, do a right rotation at the current node. For the left-right case, do a left rotation at left child of current node followed by a right rotation at the current node itself.

   3b. If Balance Factor < -1 then that means the height of the right sub-tree is greater than the height of the left sub-tree. This indicates right-right or right-left case. In this case, if balance factor of right sub-tree of current node is less than 0 then this confirms right-right case, if it is greater than 0 then this confirms

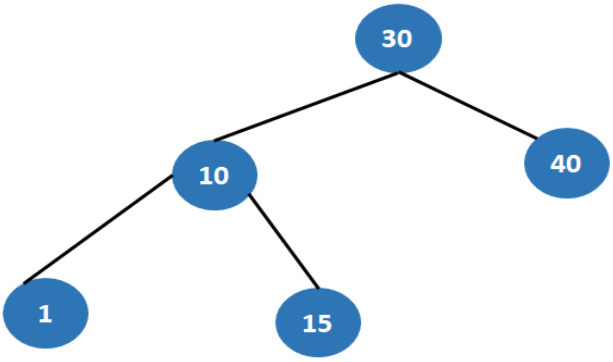
right-left case. If it is equal to 0, then we can either consider this as a right-right case or a right-left case. In this implementation, we will consider this as a right-right case. In right-right case, do a left rotation at the current node. In right-left case, do a right rotation at the right child of current node followed by left rotation at the current node itself.



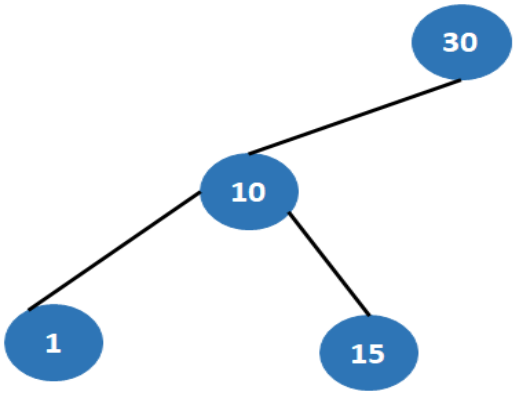
Delete 60



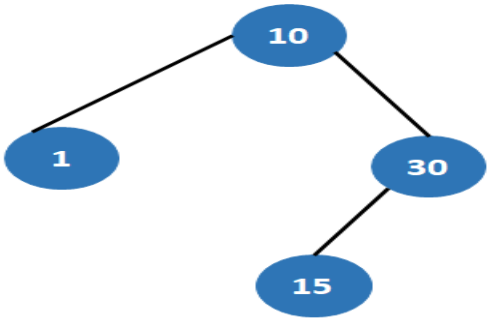
Delete 50



Delete 40



Height is not balanced at node 3, left-left case; perform the right rotation to balance the tree.



The above tree is balanced.

3. Search

Search operations returns a node if the element to be searched is present otherwise does not returns any value

Advantages of AVL Tree

1. AVL trees are height balanced trees, so the operations like insertion and deletion have low complexity.
2. provides faster search options