CSE 326
Lecture 9

AVL Trees I

Winter Quarter 1999

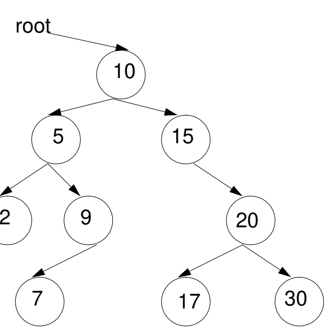
AVL Trees

- AVL trees named after Adelson-Velskii and Landis (1962) is a classic method of keeping trees relatively balanced. If there are n nodes in an AVL tree then it height is bounded by about 1.44 $\log_2 n$.
- AVL tree property:

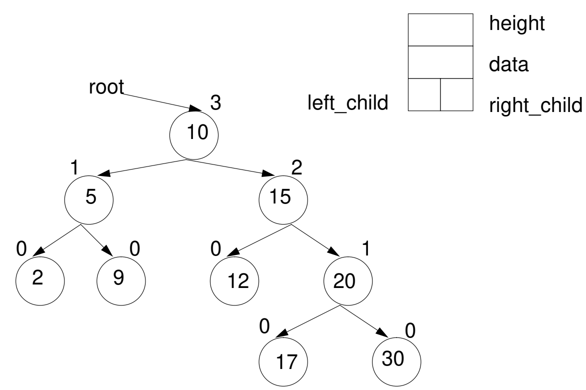
  For each node the heights of its left subtree and right subtree differ by at most 1.

Testing the AVL Tree Property

- Null's have height = -1.

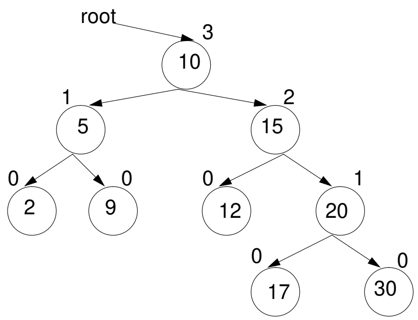


An AVL Tree

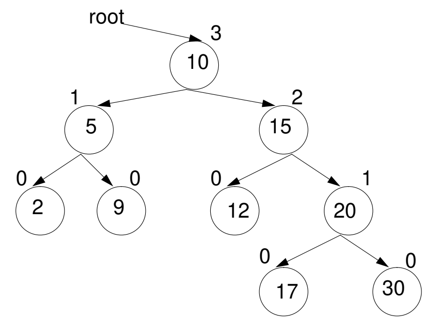


Insert 3

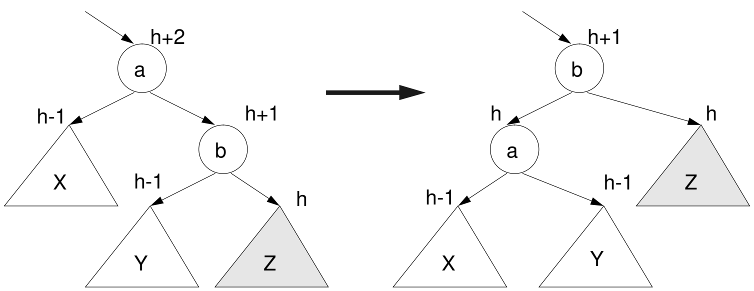
- Note that heights may change on path of insertion, but not elsewhere.



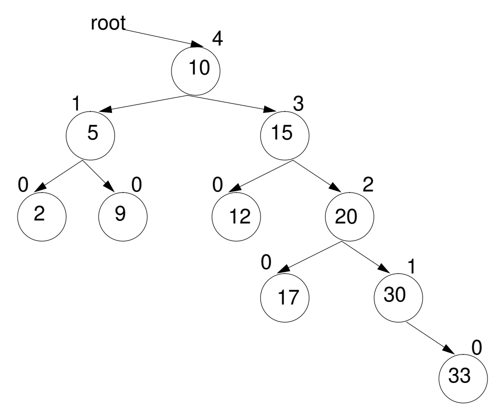
Insert 33



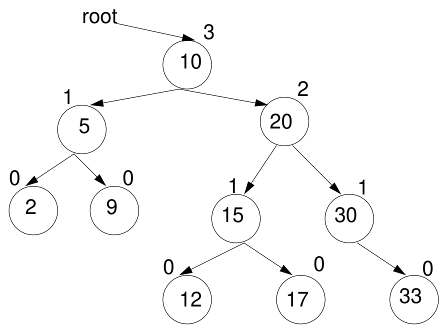
The Single Rotation



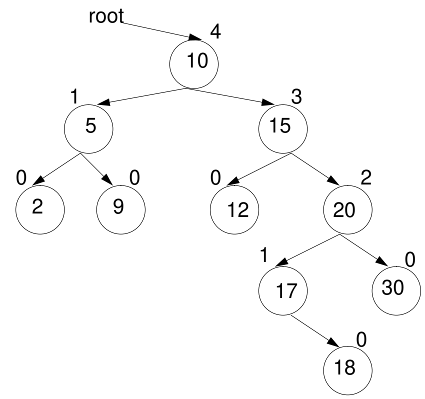
Single Rotation Example



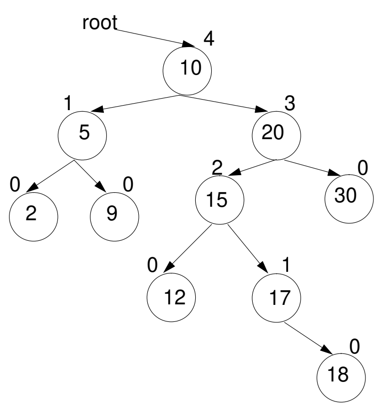
Tree After Single Rotation



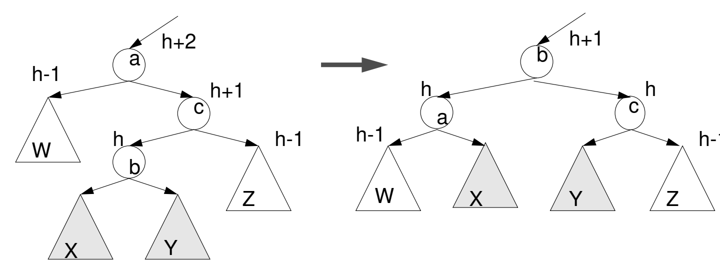
Insert 18



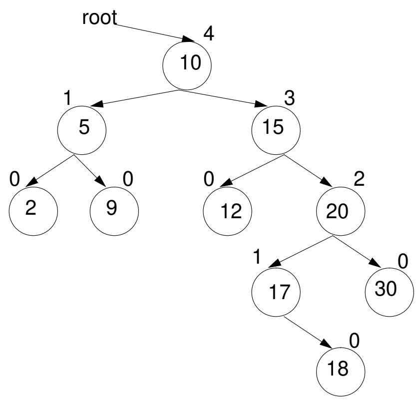
Single Rotation Won't Help



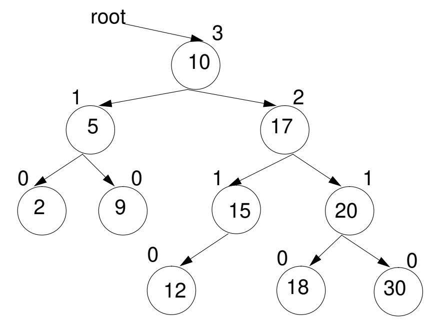
Double Rotation



Double Rotation Example



Tree After Double Rotation



Summary of AVL Insertion

- Iterative Approach

  1. Insert the new node keeping track of the insertion path in a stack.

  2. Pop the stack incrementing the height field of the nodes found until a node n is found which is not balanced or the height field does not need incrementing.

  3. If there is an imbalance then do either a single or double rotation. The rotation will be left or right depending on the direction of insertion.

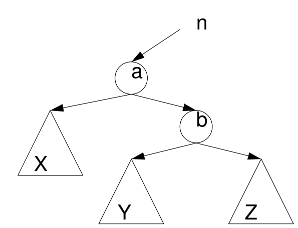
- Recursive Approach (See Weiss for very elegant solution)

  1. Recursively insert, starting at the root.

  2. On return check for imbalance of children, if imbalance then single or double rotate, otherwise update the height field. Such an imbalance will only happen once.

Single Rotation Code

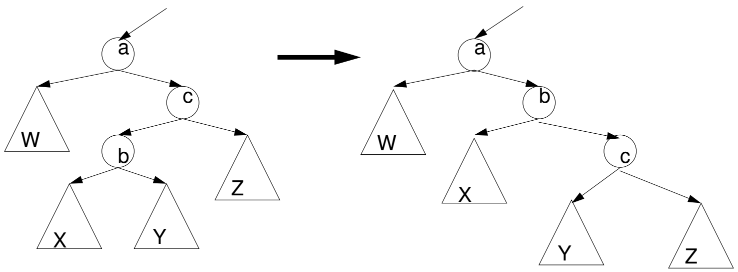
```
Single_Rotate_Right(n: reference node pointer) {
local node pointer p;
p := n.right;
n.right := p.left;
p.left := n;
n.height := max(ht(n.left),ht(n.right)) + 1;
p.height := max(ht(p.left),ht(p.right)) + 1;
n := p }
```



Right Double Rotation

- Double_Rotate_Right(n: reference node pointer) {

  Single_Rotate_Left(n.right);
  Single_Rotate_Right(n) }



Second Step in Double Rotation