Balanced Binary Search Tree

Motivation

- Runway reservation problem
- Insertion → $O(\log n)$ (Conditional)
- Binary Search Tree
- Search/Insert/Del → $O(h)$
- Unbalanced: $h = O(n)$

h: height of the tree

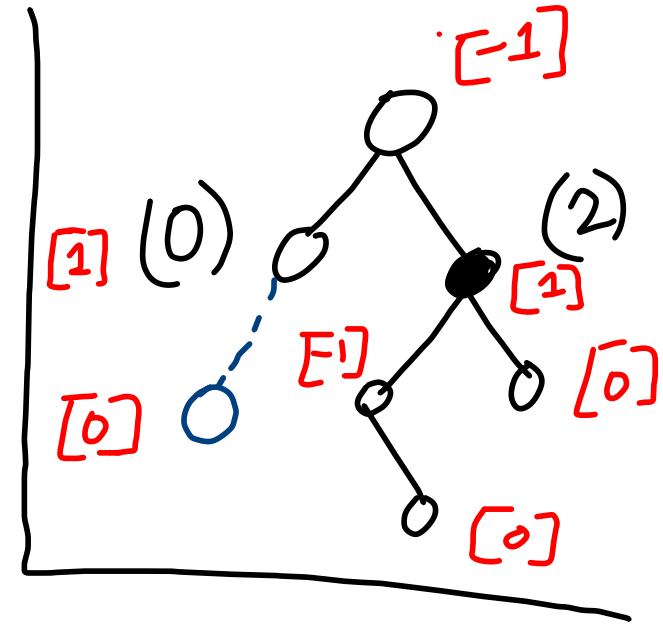
Balanced BST

Ensures $h = O(\log n)$

- AVL Tree
- Red-Black Tree

AVL Tree

(Adelson, Velsky, Landis)



height of a node in a tree : longest path from that node to any leaf

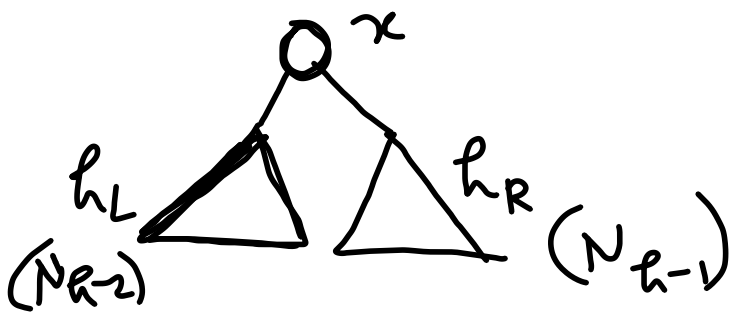
Requirement :

Height of left child & right child of any node differs by at most 1

$$= \max\{ h(\text{left child}), h(\text{right child}) \} + 1$$

[node value] → height(left sub-tree) − height(right sub-tree)

Height of Any AVL tree (with n nodes) is $O(\log n)$



$\forall x, \quad |h_L - h_R| \leq 1$

$N_h \rightarrow$ min # nodes possible with an AVL tree of height h.

$$N_h = N_{h-1} + N_{h-2} + 1$$

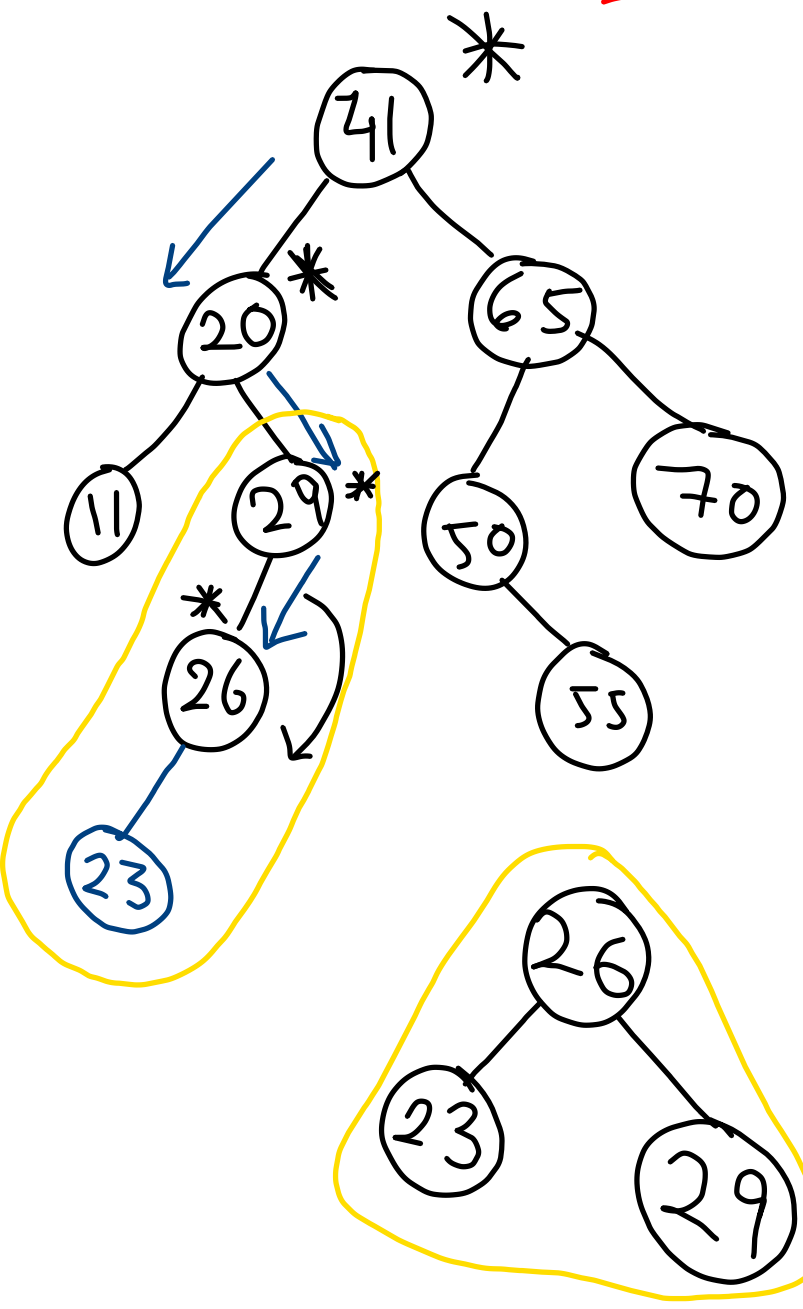
Worst Scenario:

Right Subtree has height One more than left sub-tree

$N_h > 2N_{h-2} \Rightarrow N_h = O(2^h)$

$h = O(\log n)$

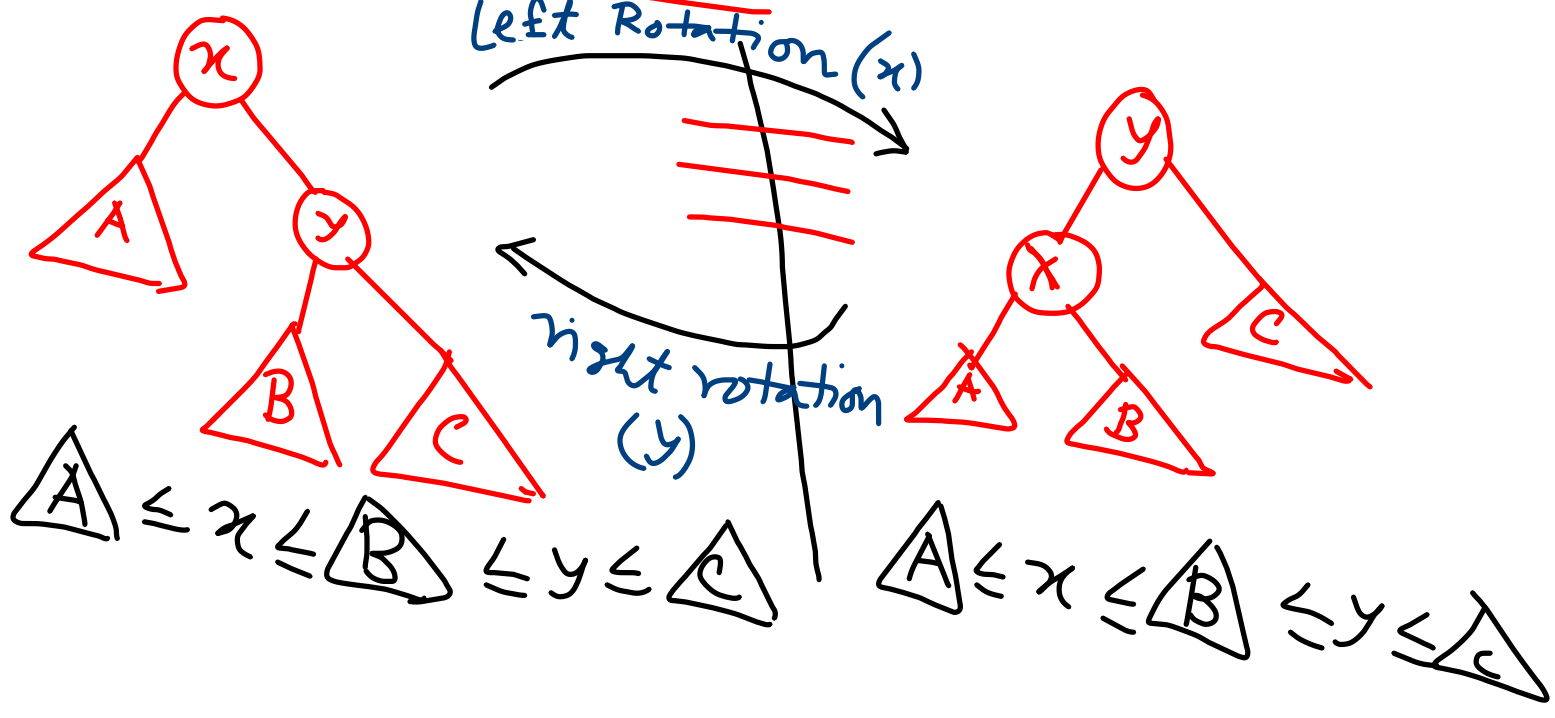
AVL Insertion



Add(23)

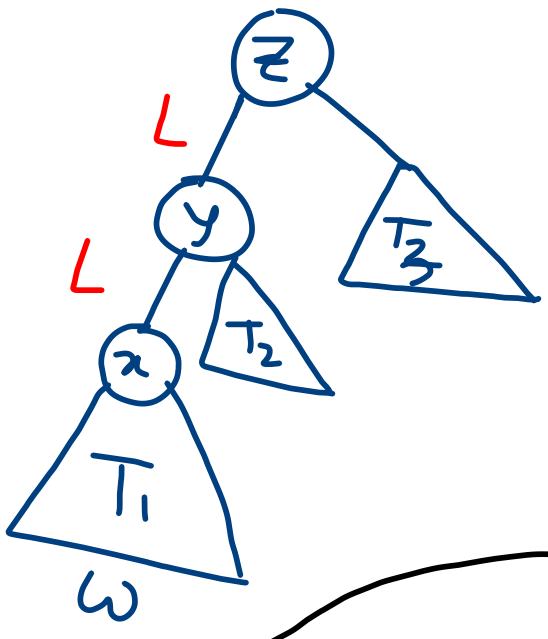
- Binary Search Insertion $O(\log n)$
- Ensuring AVL Property $O(1)$

Rotation

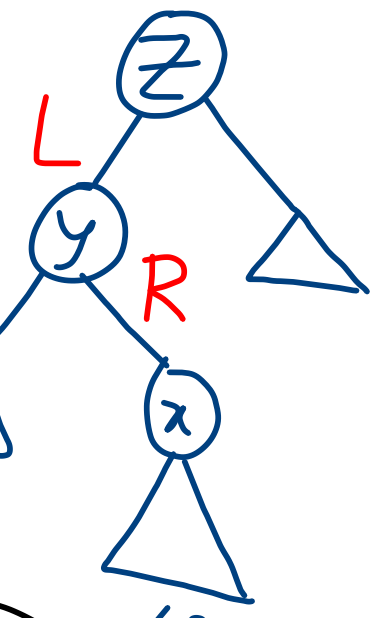


AVL Insertion (w)

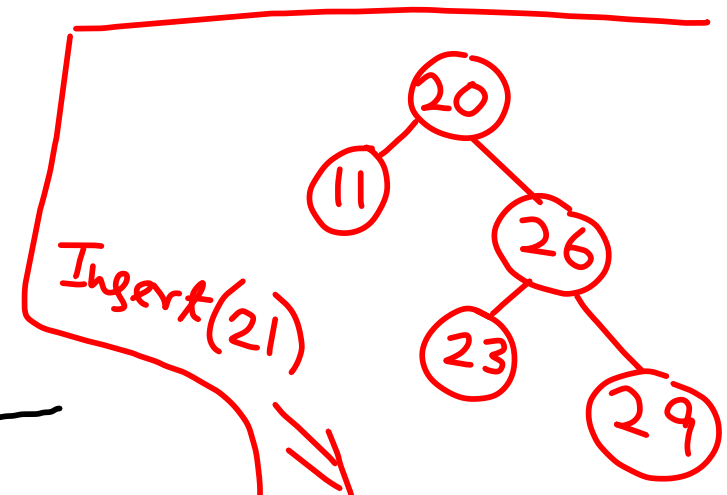
- Find out first node that do not posses AVL Property (z node)
- Grand child of z which is in the path z → w (node x)

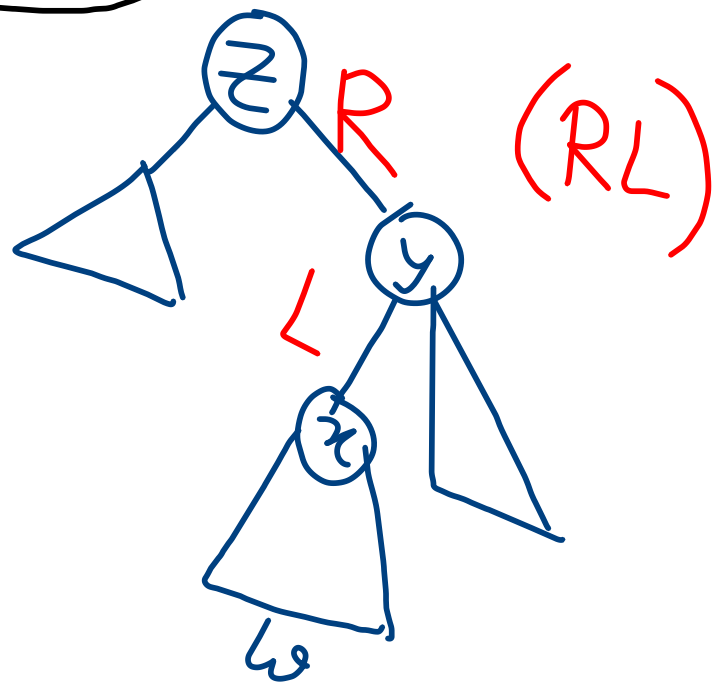


(LL)

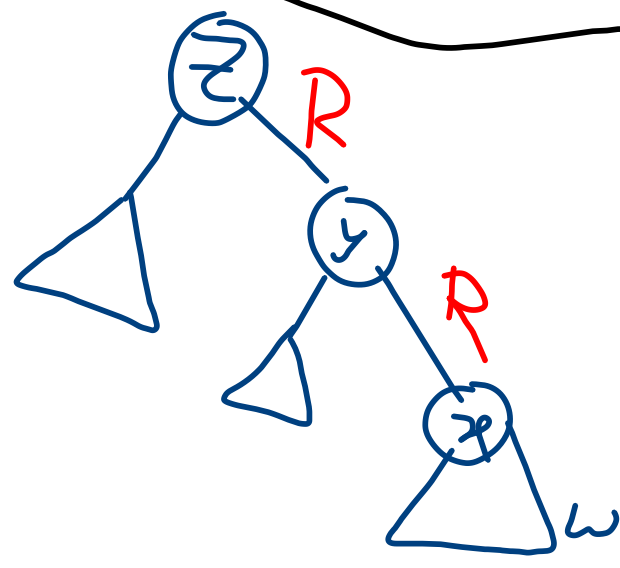


(LR)

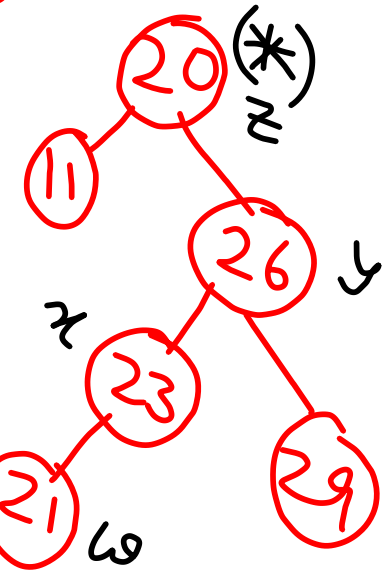




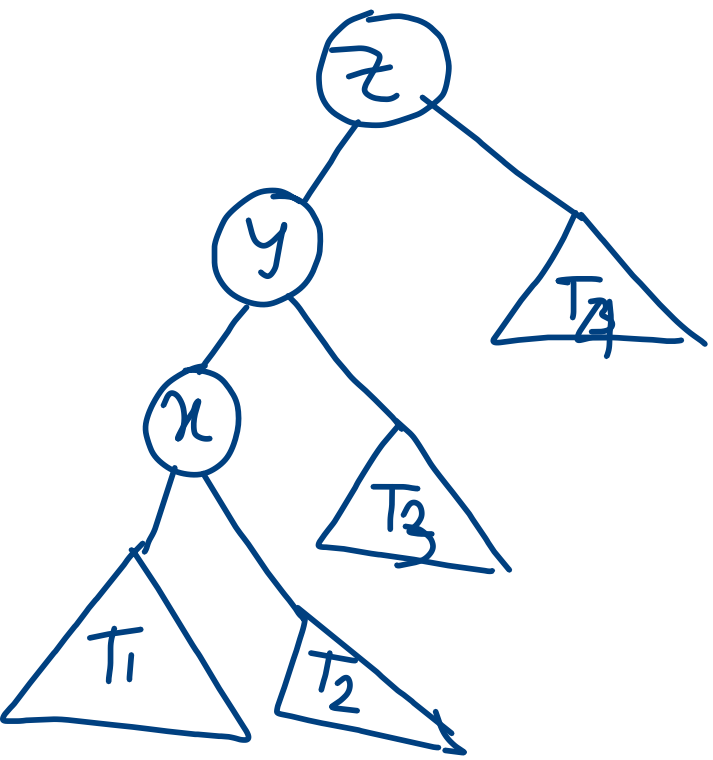
(RL)



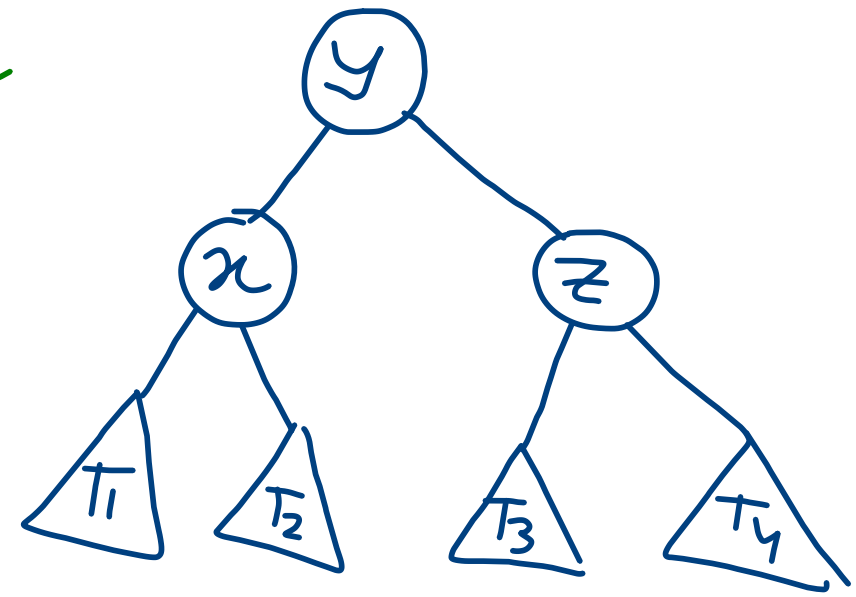
(RR)



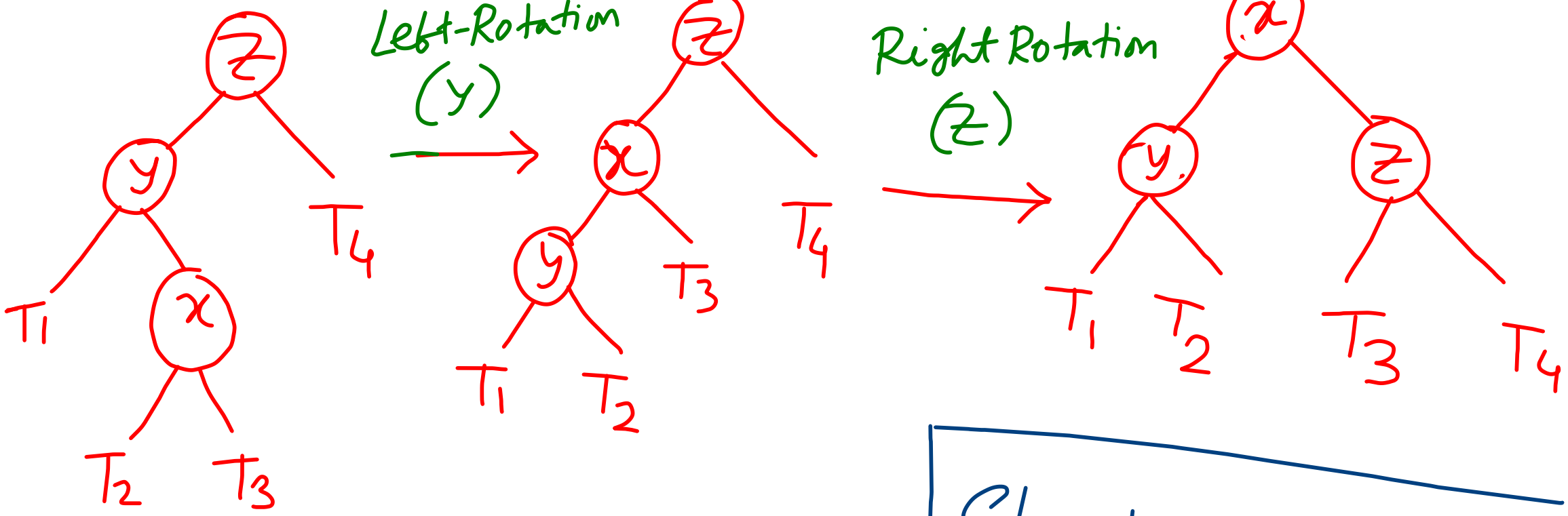
(L L)



Right Rotation



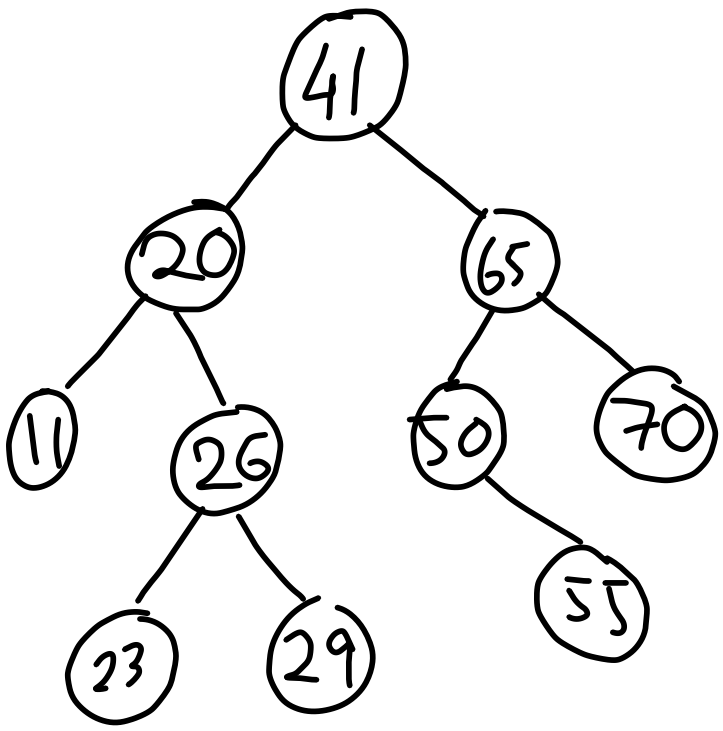
(LR)



$T_1 \le y \le T_2 \le x \le T_3 \le z \le T_4$

Check for RL & RR Rotation

What happens while Deletion ?



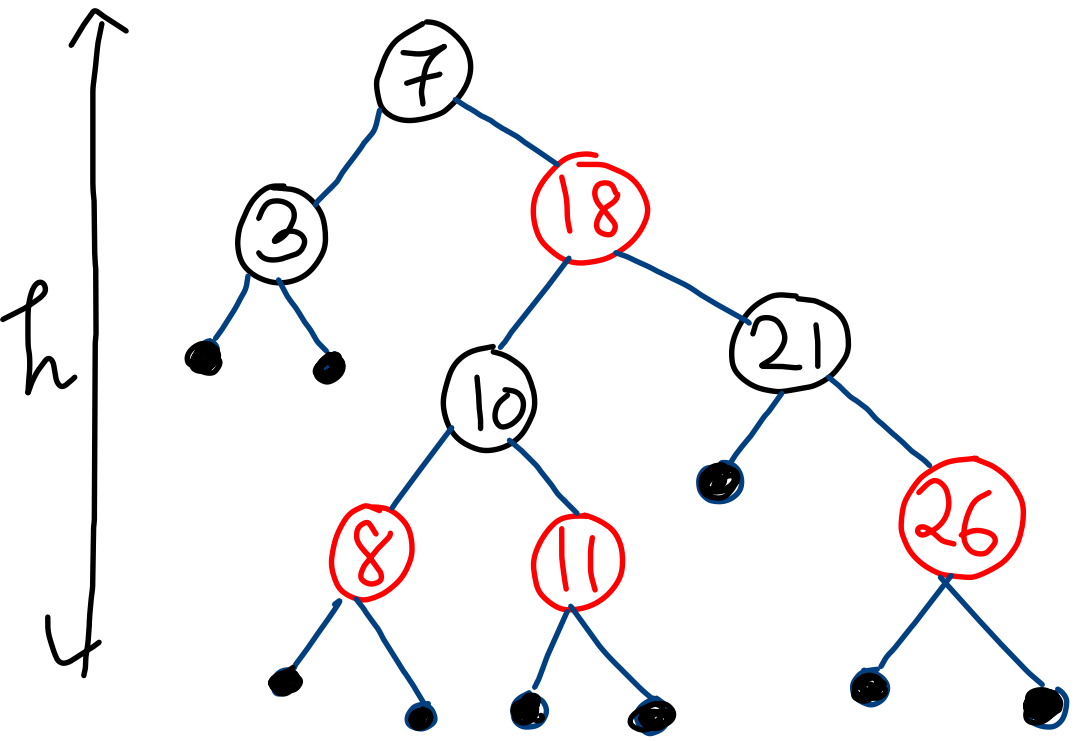
1. Delete(65)
2. Delete(20)
3. Delete(41)

Red-Black Tree

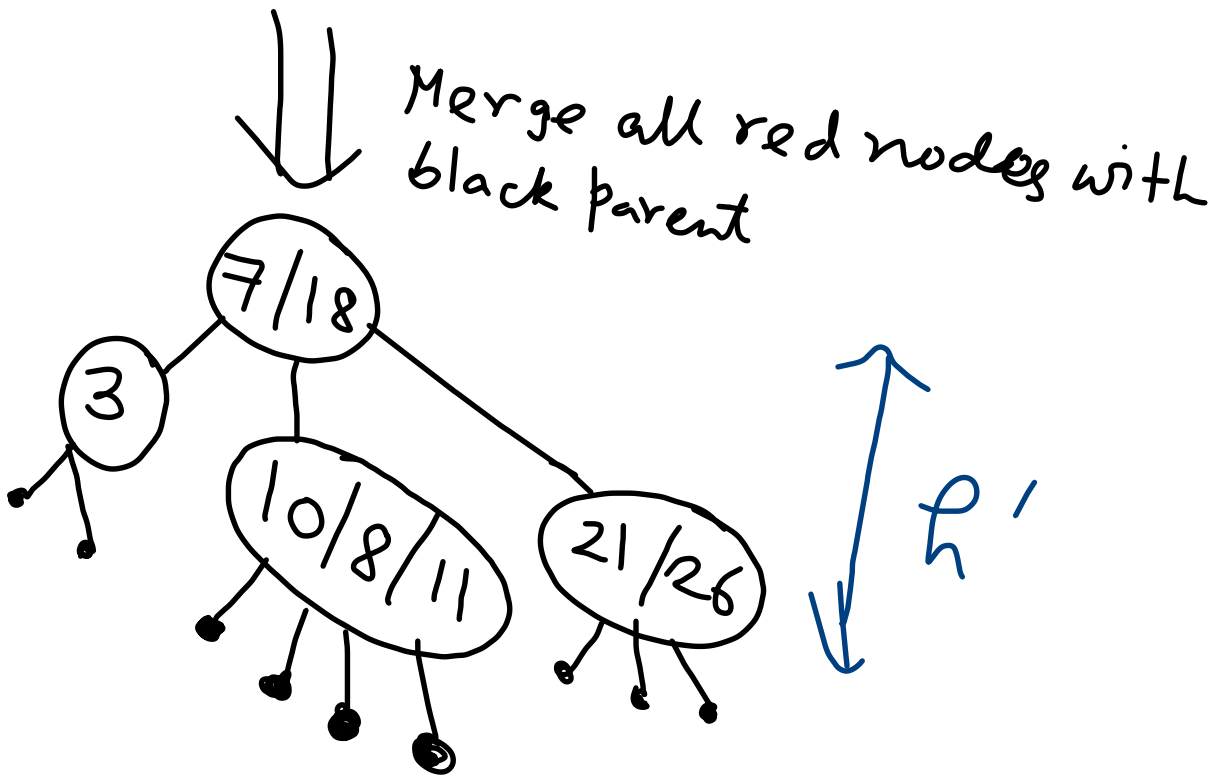
Binary Search Tree with colors (red, black)

Properties

1) Each node → either red or black
2) The root & the leaves (NIL Pointers) are black
3) Every red node has black parent.
4) All simple paths from node x to a leaf have same no. of black nodes.



Height of Red-Black Tree is $O(\log n)$



If Merged

- ↳ Each node has 2, 3, 4 children (2-3-4 tree)
- ↳ Depth is same for all nodes.
- ↳ #leaf = $n+1$ (Prove it)

$$2^{h'} \le n+1 \le 4^{h'}$$

$$h' \le \log(n+1)$$

$$h \le 2h' \le 2\log(n+1)$$