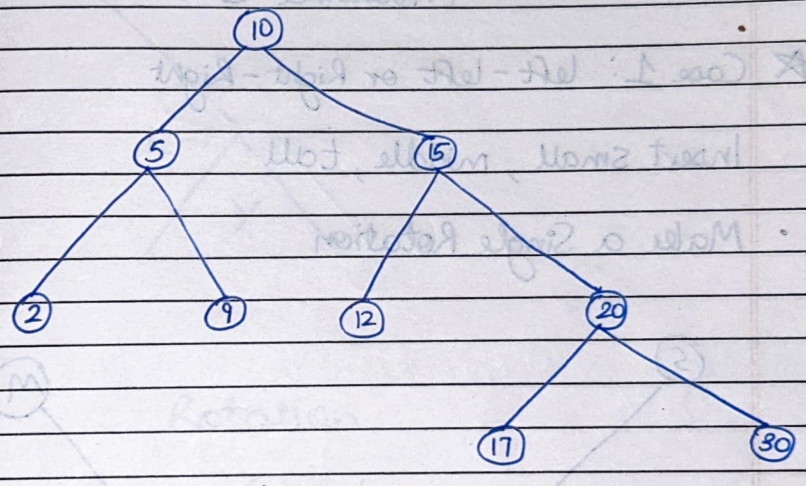
Lecture 6
AVL Trees

Balance == Height of left subtree − Height of right subtree

Perfectly Balanced → Zero
Small Everywhere → Balance Enough



Considering leaf node height = 0

Every node must be balanced

Good Write

DATE : / /
PAGE :

Good Insert Case :

Insert middle, then small, then Tall

Eg.

h

h

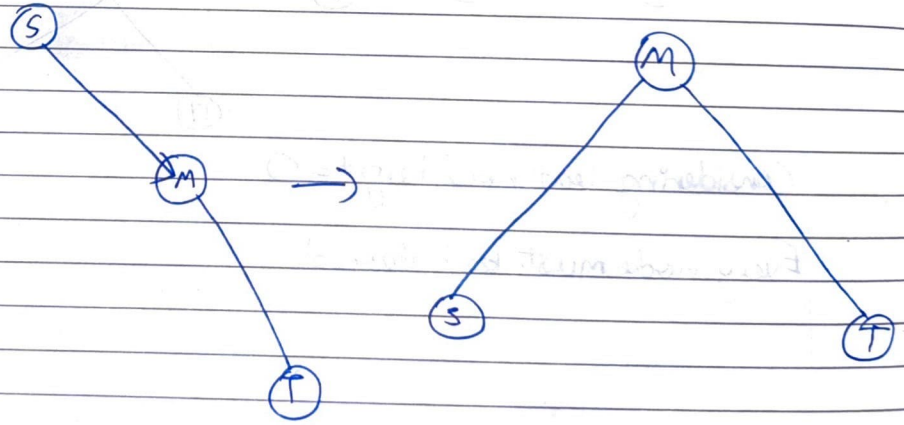
Z

Imbalance Cases

★ Case 1: left-left or Right-Right

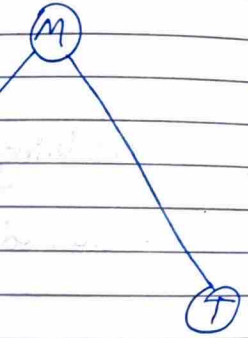
Insert small, middle, tall

- Make a Single Rotation



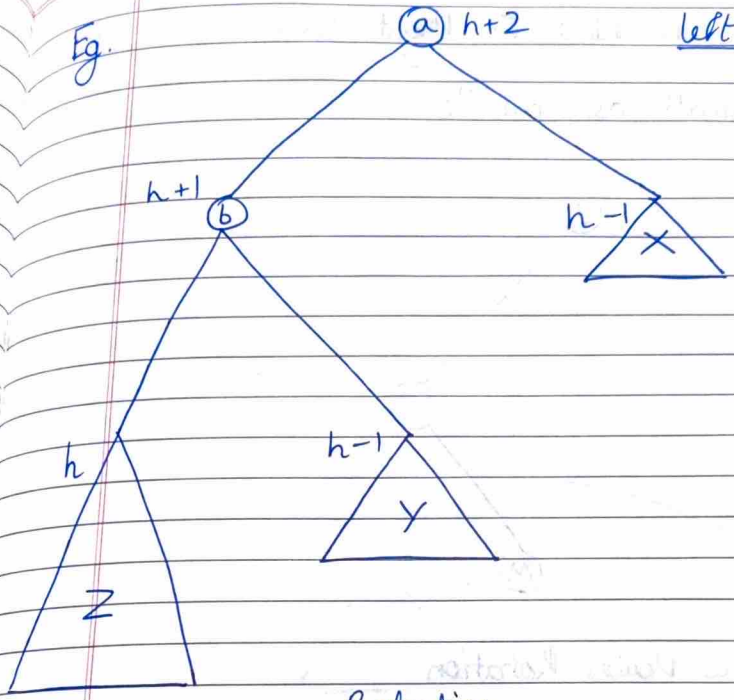
A right child can have its parent as its left child.

Tall



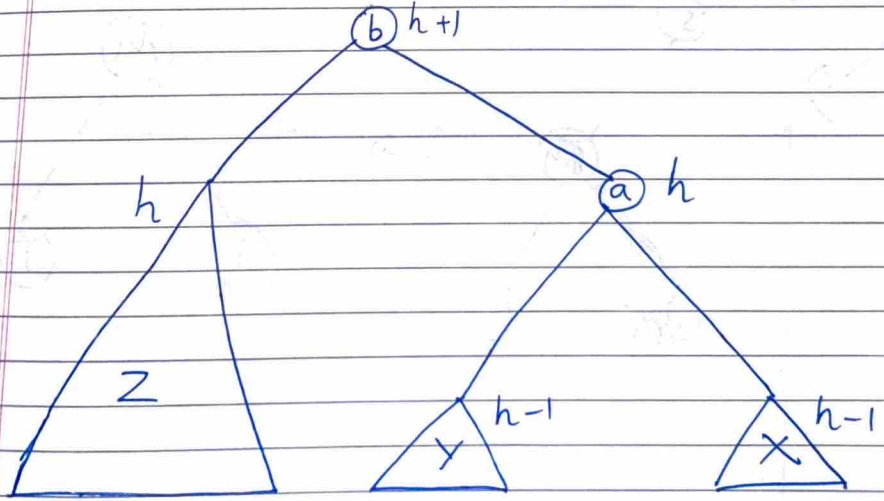
its left child.

Eg.



Left - Left Imbalance

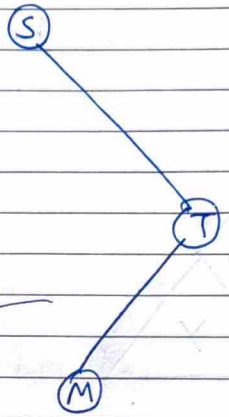
Rotation



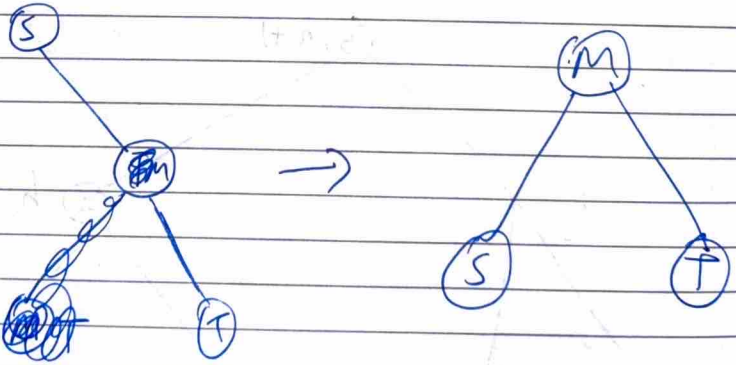
DATE : / /
PAGE :

★ Case 2 : Left - Right or Right - left

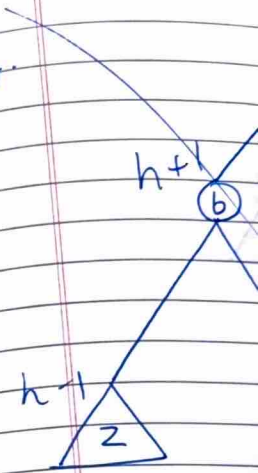
Insert small, tall, middle



- Requires Double Rotation

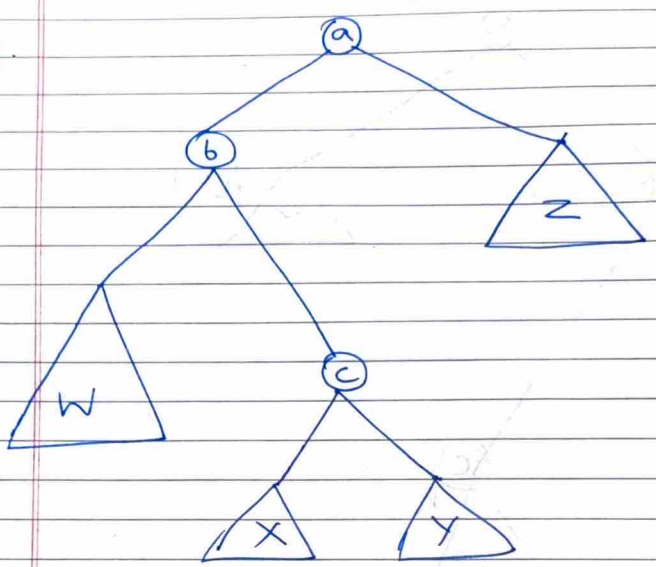


Eg.



DATE: __/__/____
PAGE: _____

Eg.



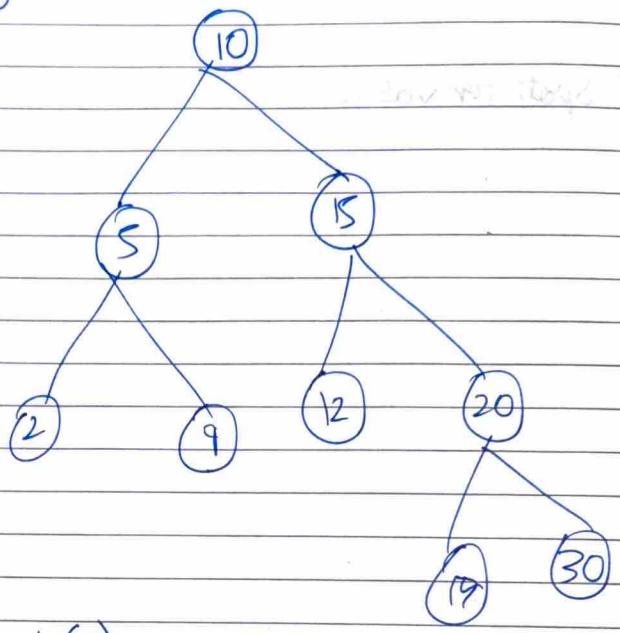
Double Rotation

Find Sp

DATE : __/__/__
PAGE :

Easy Insert :

Eg.

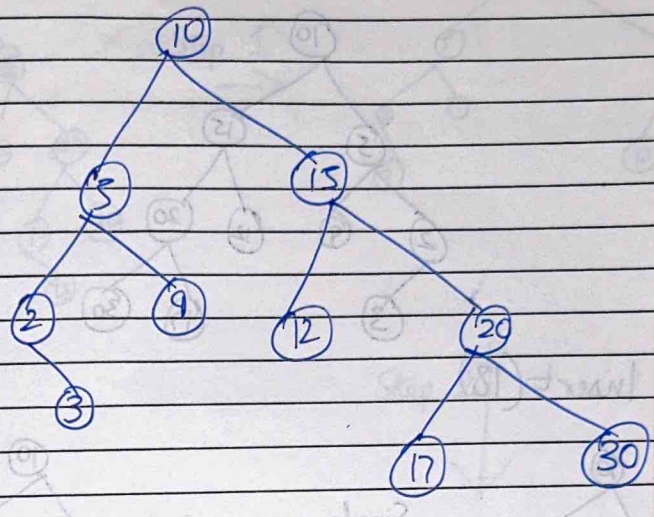


Insert (3)

Hard

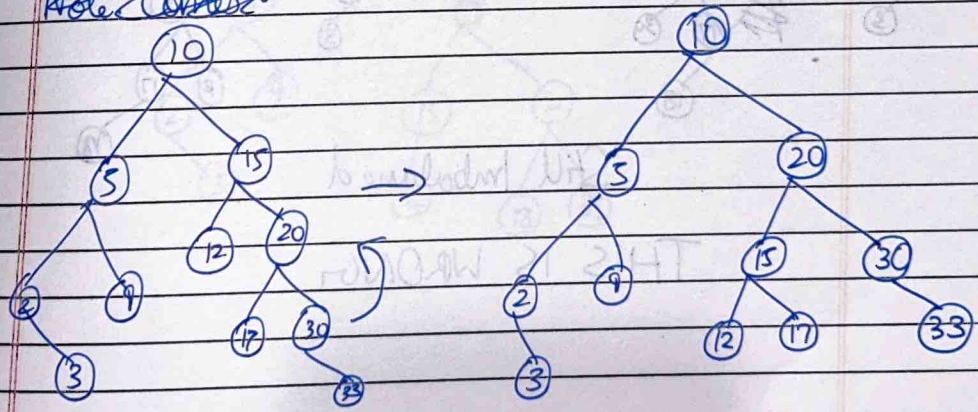
In

Hard Insert : (Bad Case 1)



Insert (33)

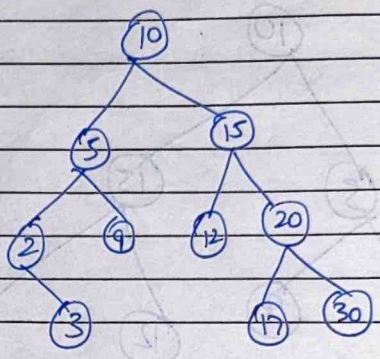
~~Note: Comes~~



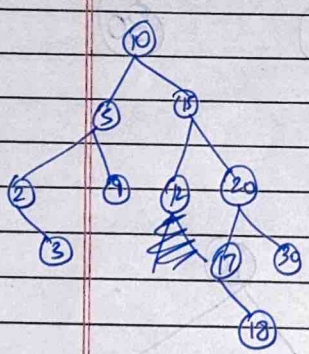
Good Write

DATE: __/__/____
PAGE: _____

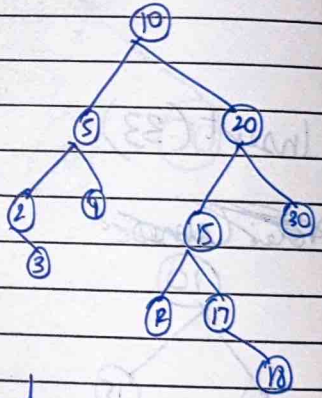
Hard Insert : (Bad Case 2)



Insert (18)

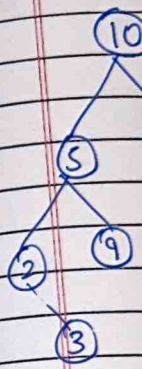


Single Rotation →



Still Imbalanced

THIS IS WRONG

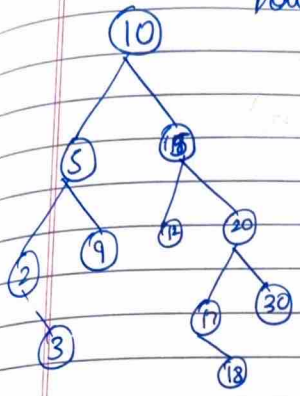


Good Write

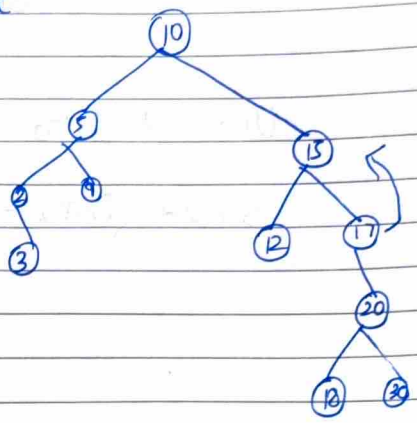
DATE : __/__/____
PAGE : _____

DATE : __/__/____
PAGE : _____

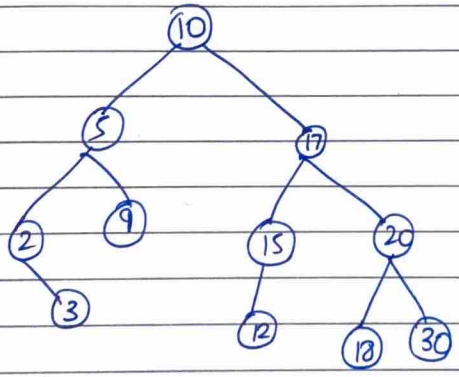
Double Rotation



Step 1 →



Step 2 ↓



DATE: __/__/____
PAGE: _____

Building AVL Tree

Unsorted Data → $\Theta(n \log n)$

Sorted Data → $\Theta(n)$