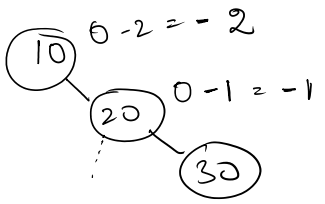
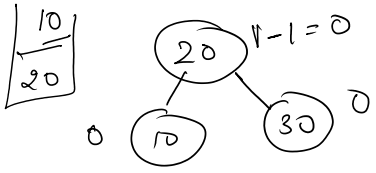
AVL trees – Height Balanced Trees

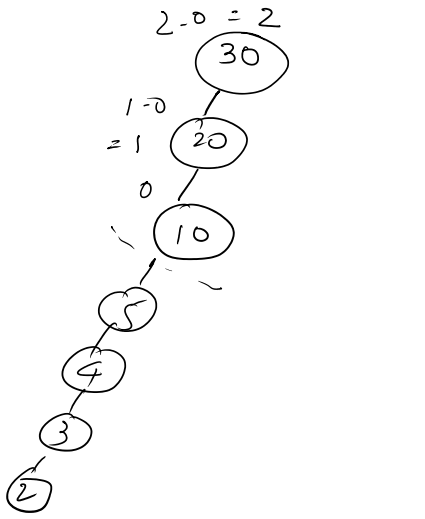
#9 10, 20, 30



20, 10, 30



30 20 10



Balance factor

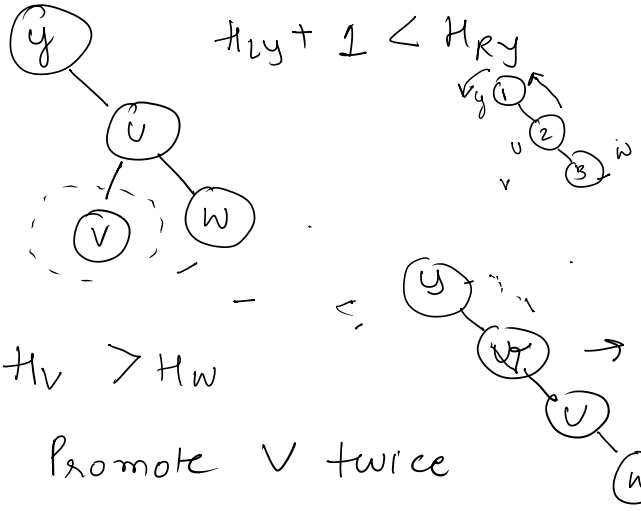
$= \{Hgt\ L - Hgt\ R\} = \{-1, 0, 1\}$

Insertion in AVL Trees

1. Insert 'x' using BST insertion algo, pushing the visited nodes into a stack
2. Check if stack is empty
   i) If empty ⇒ Tree is balanced. Stop
   ii) Remove 'y' – top of the stack
   iii) Compute if $(H_{Ly} - H_{Ry}) = \{-1, 0, 1\}$
   Goto Step 2
   else
   Check the case in which 'y' falls & balance accordingly

Case I

BF < 0 → $H_{left\ ST} - H_{right\ ST}$



If $H_v > H_w$

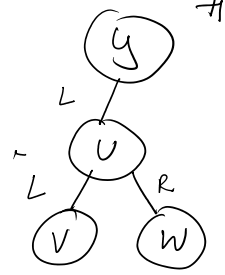
Promote V twice

else $H_w > H_v$

Promote U

Case II BF > 0

$H_{Ly} > H_{Ry} + 1$



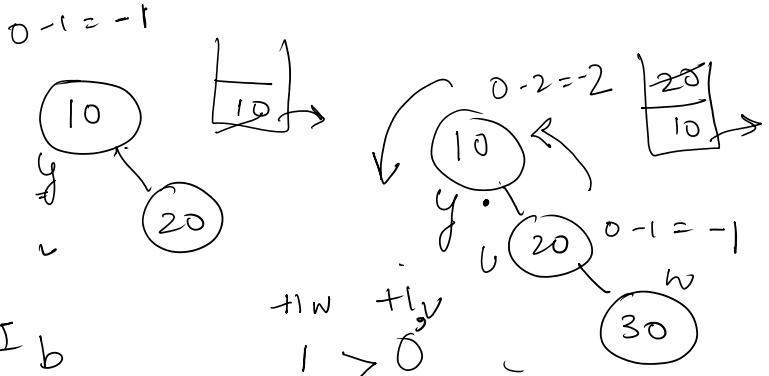
if $H_v > H_w$

Promote U

else $H_w > H_v$

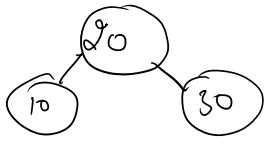
Promote w twice

#q 10, 20, 30

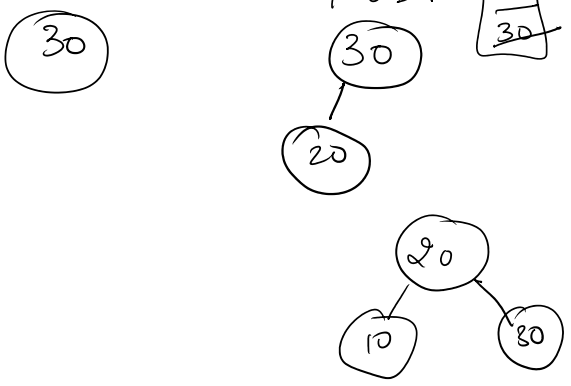


20, 10, 30

Case Ib



#q 30, 20, 10



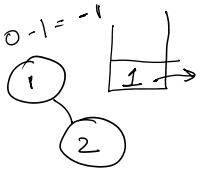
Promote 20

#q Insert 1, 2, 3, 4, 5

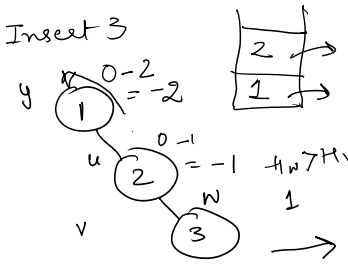
#g Insert 1, 2, 3, 4, 5, 0

Insert 1

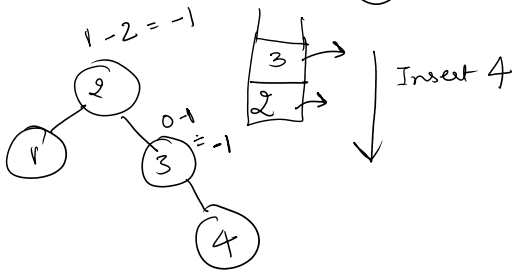
Insert 2



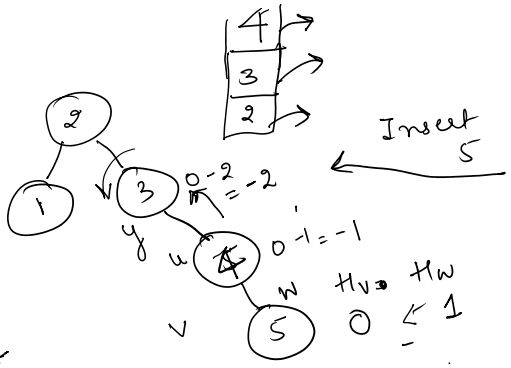
Insert 3



Insert 4

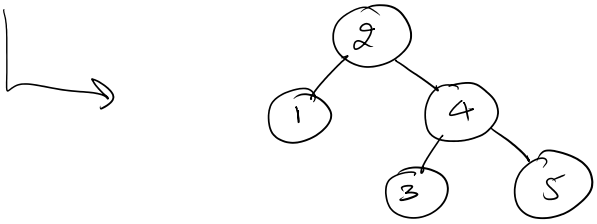


Insert 5

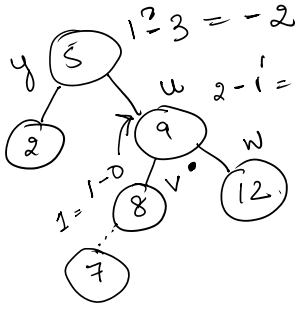


Case I

Promote '4'

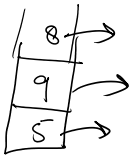


#g



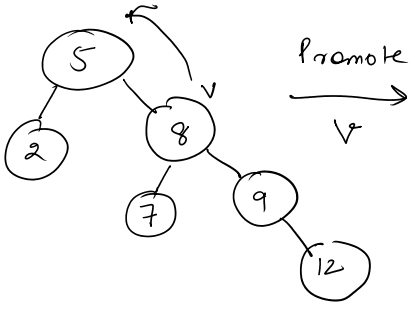
Promote V

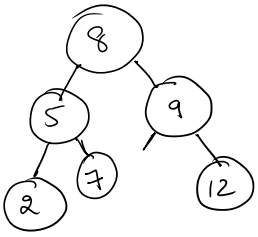
Insert 7



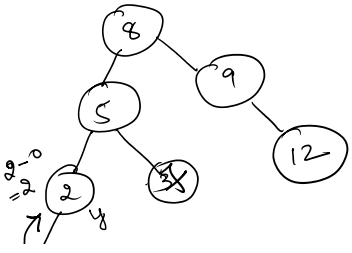
hv hw
2 > 1

Promote v twice

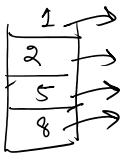


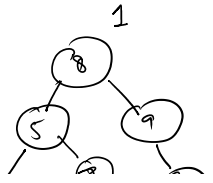


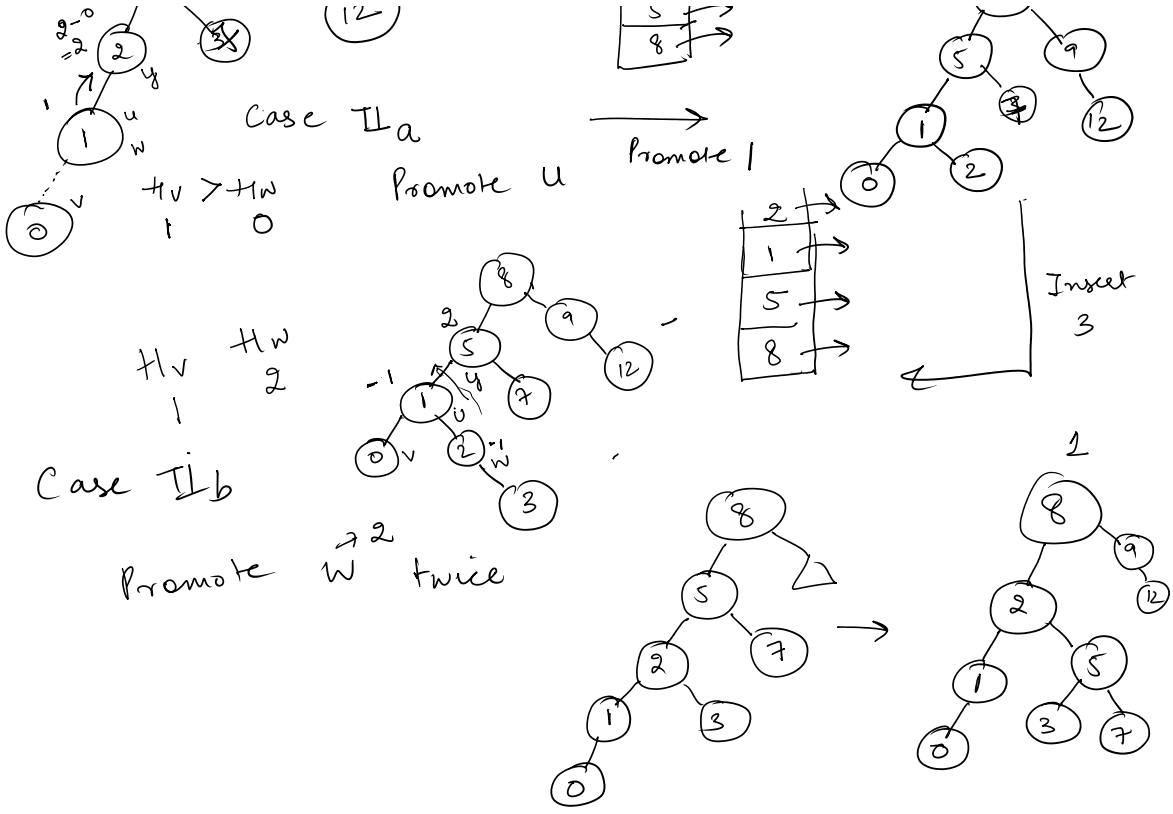
#g



Insert 0



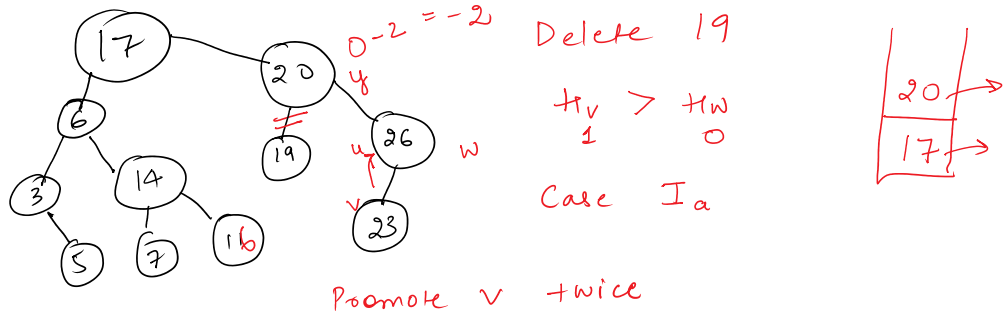




Deletion in AVL trees

- Delete the node as we do BST
- Push the visited nodes in the stack
- While stack is not empty
  - Pop item
  - Balance the tree (BF, y U VW, Case I/II)
  - Continue till stack is empty.

#9



Promote v twice