C-Programming and DS

Today Class Topics

BST
AVL-Tree

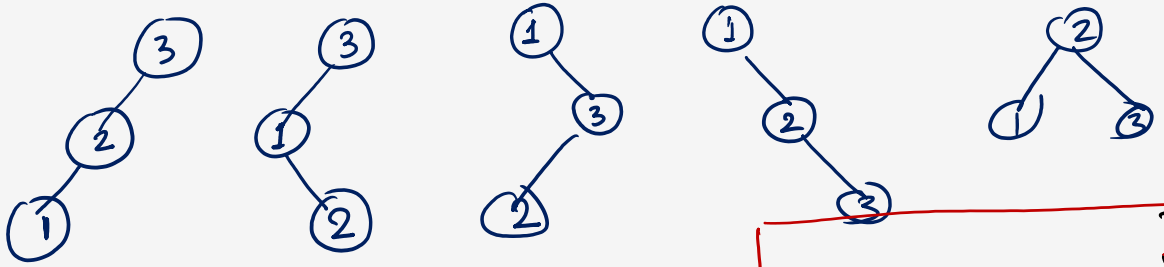
Binary Search tree (BST) :-

No. of Binary search trees :-

1) No of BST = No of Unlabeled Binary Tree of n keys = $\frac{{}^{2n}C_n}{n+1}$

if $n = 3$, then of BST = $\frac{{}^{6}C_3}{4} \Rightarrow \frac{6!}{3!\,3! \times 4} \Rightarrow \frac{6 \times 5 \times 4}{6 \times 4} = 5$

Let 1, 2, 3



2) b_n is the No. of BST of n nodes

$$b_n = \frac{{}^{2n}C_n}{n+1} = \sum_{k=0}^{n-1} b_k \, b_{n-1-k}$$

Let T(n) be the number of different binary search trees on n distinct elements.

$$T(n)=\sum_{k=1}^{n} T(k-1)\,T(x)$$

Then , where x is

- [ ] n-k+1
- [x] n-k
- [ ] n-k-1
- [ ] n-k-2

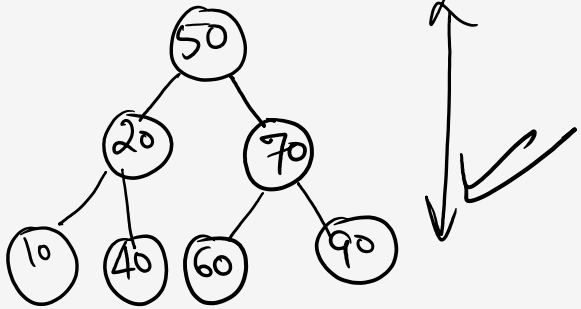
$$b_n = \sum_{k=0}^{n-1} b_k\, b_{n-1-k}$$

$$= b_0\, b_{n-1} + b_1\, b_{n-2}$$

$$= T(0)\,T(n-1) + T(1)\,T(n-2)$$

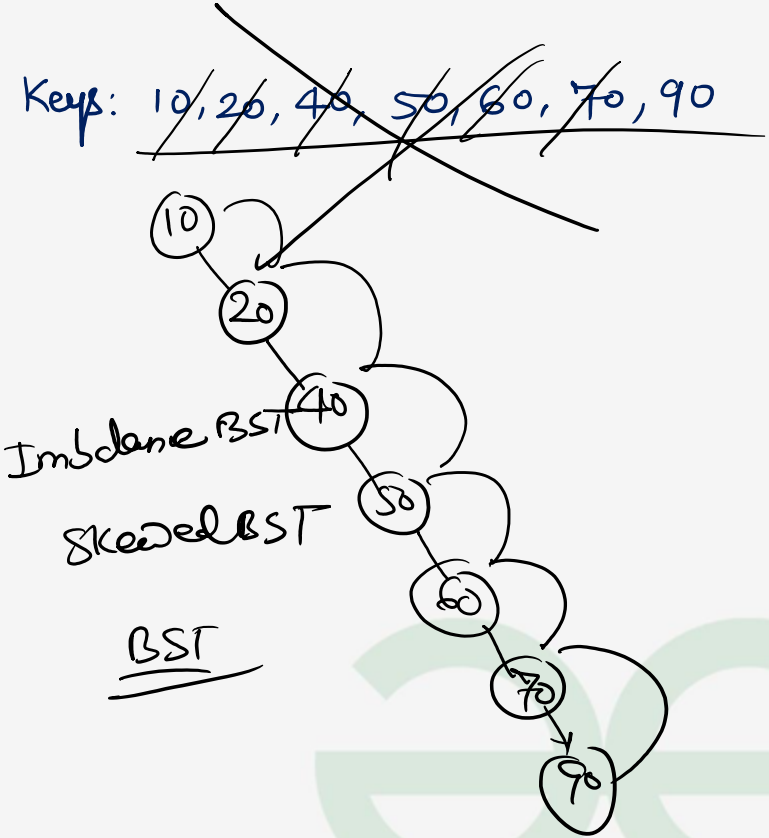
Binary Search Tree Construction :-

Keys: 50, 70, 60, 20, 90, 10, 40



BBST

Keys: 10, 20, 40, 50, 60, 70, 90



Imbalance BST

Skewed BST

BST

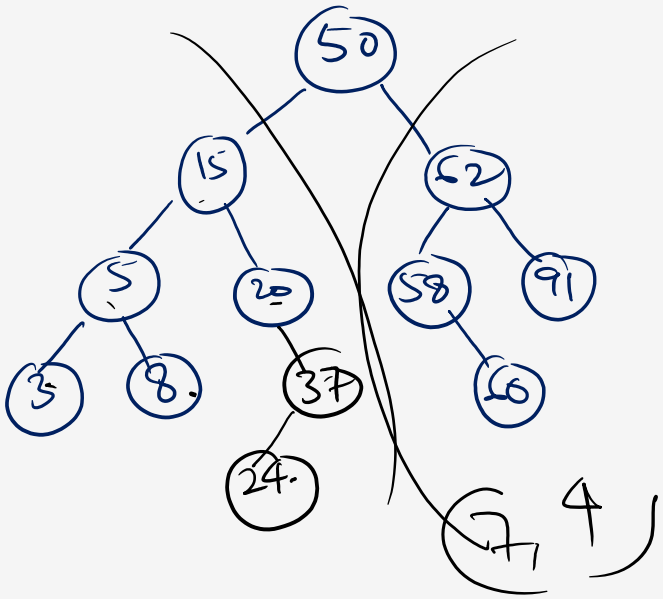
Question: A binary search tree is generated by inserting in order of the following integers 50, 15, 62, 5, 20, 58, 91, 3, 8, 37, 60, 24, The no. of nodes in the Left Subtree and right Subtree of the root respectively is ____

a) (4,7)

b) (7,4)

c) (8,3)

d) (3,8)



Question:- How many distinct Binary search trees can be Constructed out of 4-distinct keys

a) 5

b) 14

c) 24

d) 35

Binary search trees ⇓ Unlabeled Binary tree

$$= \frac{{}^{2n}C_n}{n+1} \Rightarrow \frac{{}^{8}C_4}{5}$$

Question:- We are given a set of n-distinct elements and an unlabeled binary tree with n-nodes. In how many ways can we populate the tree with given set so that it becomes a binary search tree?

a) 0

b) 1

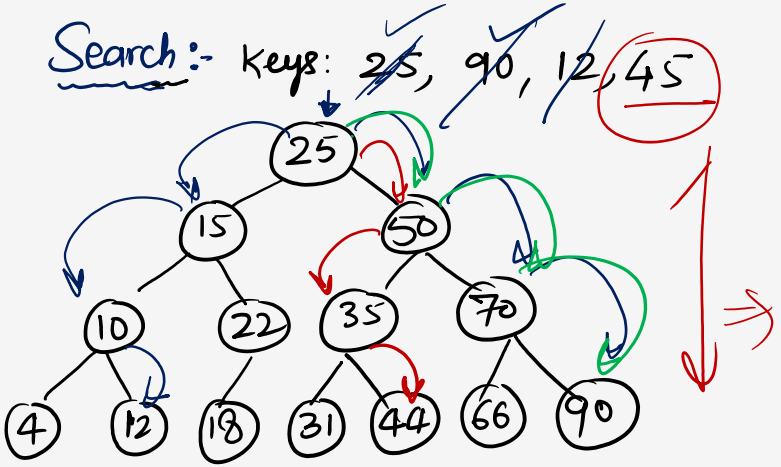
c) $n!$

d) $\frac{2n_{C_n}}{n+1}$

Binary search tree operation:

- Search ✓
- Insert ✓
- Delete ✓

Search :- Keys: ~~25~~, ~~90~~, ~~12~~, 45

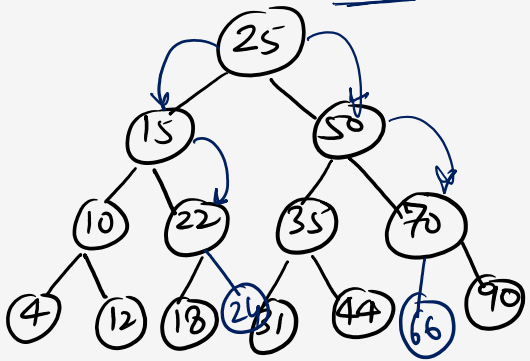


TC :-

BC = O(1)

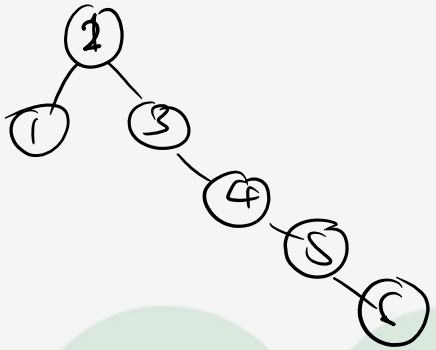
WC = O(n)

Insert :- 24, 66



BC = O(1)

WC = O(n)



TC:

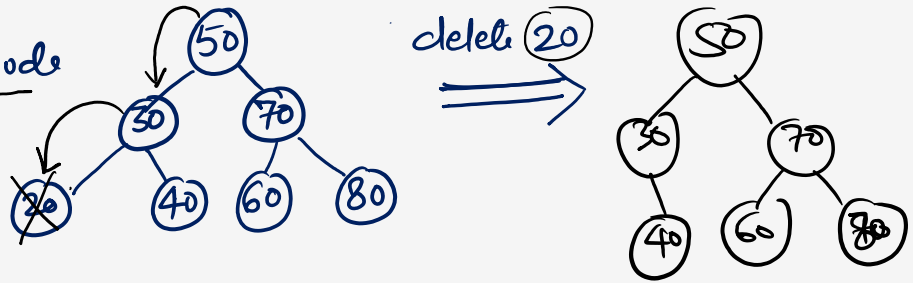
SearchTime — O(h)

AddLink — O(1)

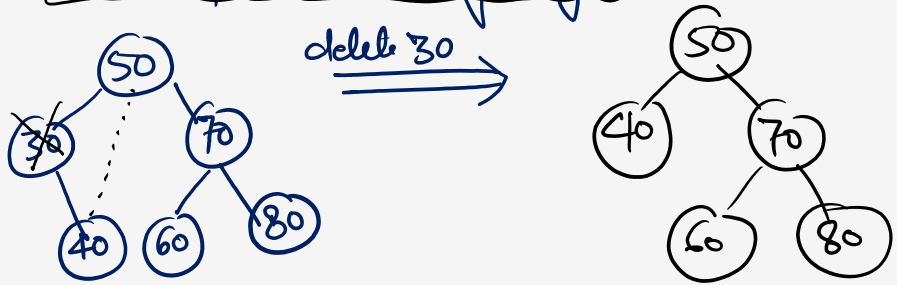
O(h)

Delete :- When it comes to deleting a node from the binary search tree, following three cases are possible

Case 1 :- Delete Leaf node



Case 2 : Deletion of a node having only one child



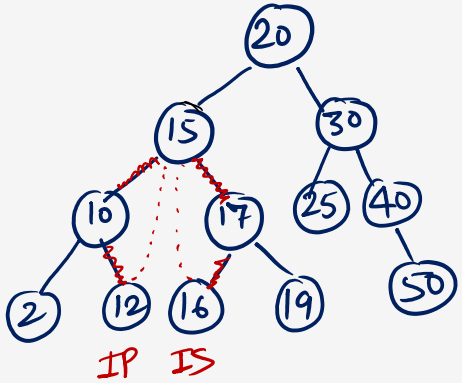
Case 3:- Deletion of a node having two childres

Delete

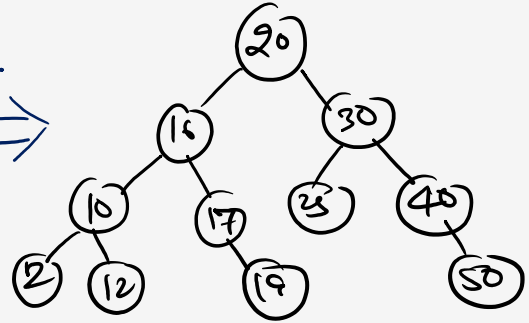
TC = O(h)

WC = O(h)

BC = O(1)



delete 15



Inorder: 2 10 12 15 16 17 19 _____

Question: Preorder of BST = 30, 20, 10, 15, 25, 23, 39, 35, 42

Postorder Traversal = ?

a) 10, 20, 15, 23, 25, 35, 42, 39, 30

b) 15, 10, 25, 23, 20, 42, 35, 39, 30

c) 15, 20, 10, 23, 25, 42, 35, 39, 30

d) 15, 10, 23, 25, 20, 35, 42, 39, 30

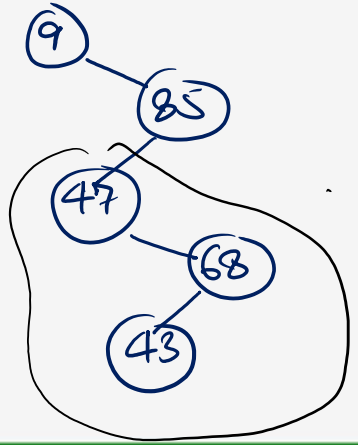
Question:- (1-100) in BST search for 55 which of the following sequence can not be the sequence of nodes examined?

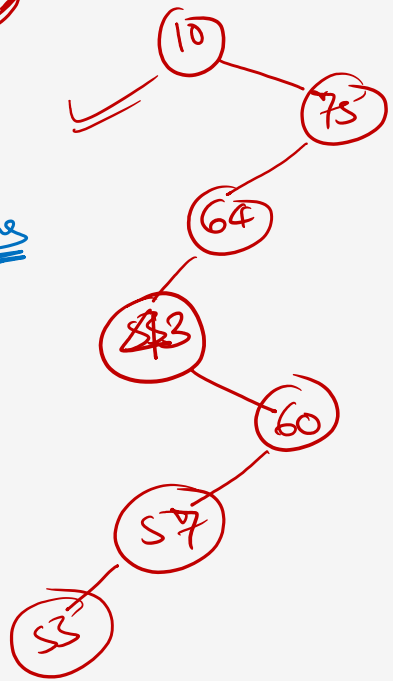
a) {10, 75, 64, 43, 60, 57, 55}

b) {90, 12, 68, 34, 62, 45, 55}

c) {9, 85, 47, 68, 43, 57, 55} ✗ Ans

d) {79, 14, 72, 56, 16, 53, 55}





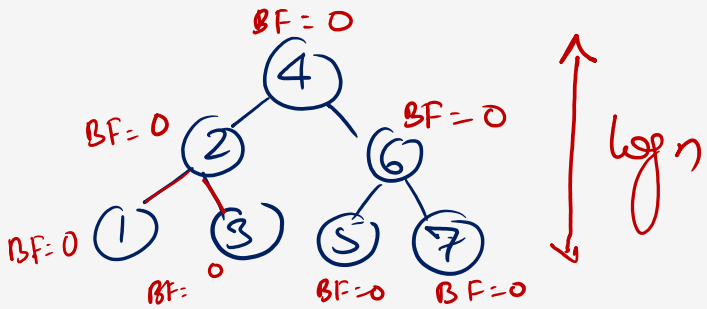
Height balance tree (AVL-tree) :-

Search, Insert, Delete → $O(h) \Rightarrow O(\log n)$ ✓

- It is a BST, in which Balance factor of each node is $\boxed{0, 1, -1}$

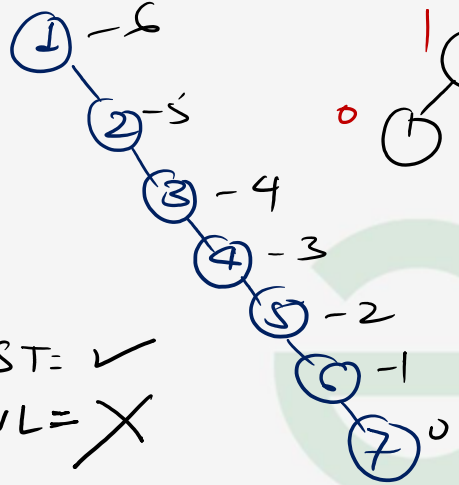
Balance factor of node $\boxed{(BF) = H(LST) - H(RST)}$

$BF = 0, 1, -1$



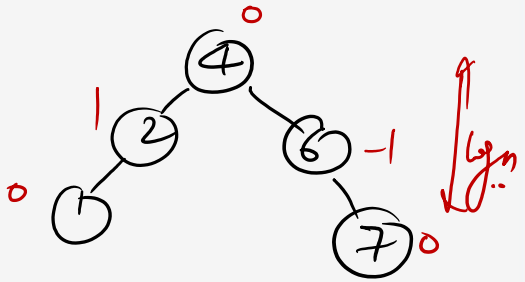
BST ✓

AVL ✓



BST: ✓

AVL: ✗



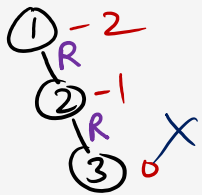
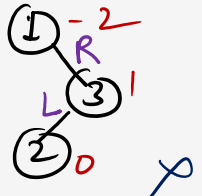
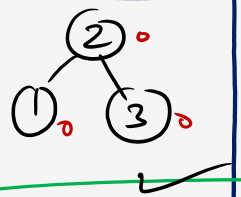
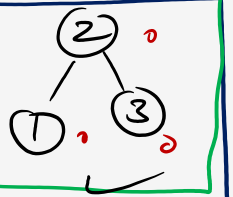
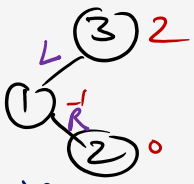
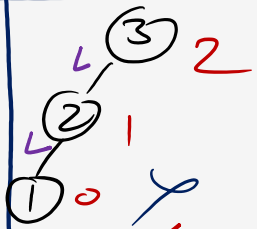
BST ✓

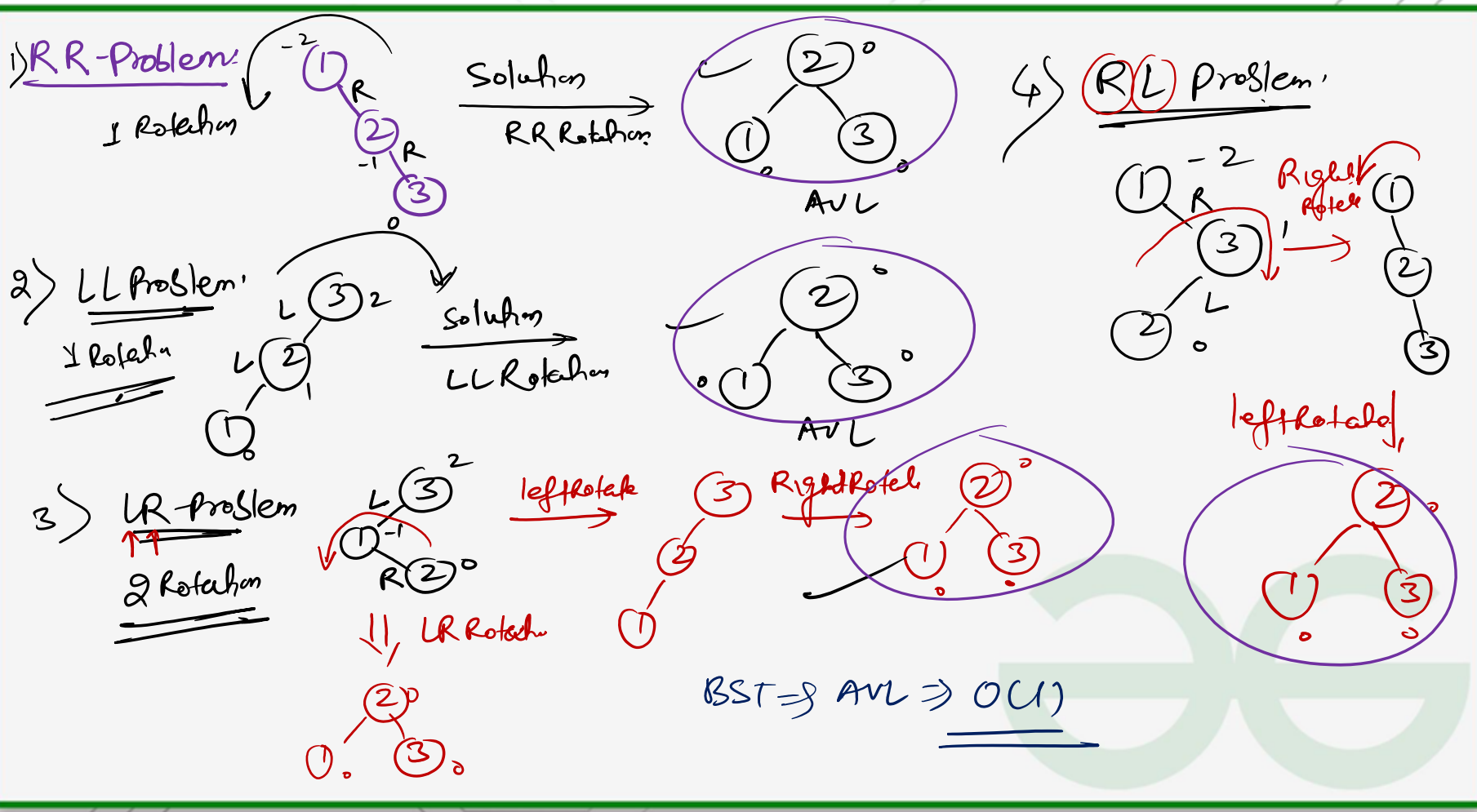
AVL ✓

Why AVL-tree:-

Because BST worstcase = O(n)
AVL Tree worstcase = O(log n)

If n = 3 (1, 2, 3)

like: (1 2 3)	(1 3 2)	(2 1 3)	(2 3 1)	(3 1 2)	(3 2 1)
					
BST ✓ AVL ✗	BST ✓ AVL ✗	BST ✓ AVL ✓	BST ✓ AVL ✓	BST ✓ AVL ✗	BST ✓ AVL ✗


1) RR-Problem
1 Rotation
Solution
RR Rotation
AVL
2) LL Problem
1 Rotation
Solution
LL Rotation
AVL
3) LR-Problem
2 Rotation
leftRotate
RightRotate
II, LR Rotation
4) RL Problem
RightRotate
leftRotate
BST ⇒ AVL ⇒ O(1)

AVL-Tree operation:-

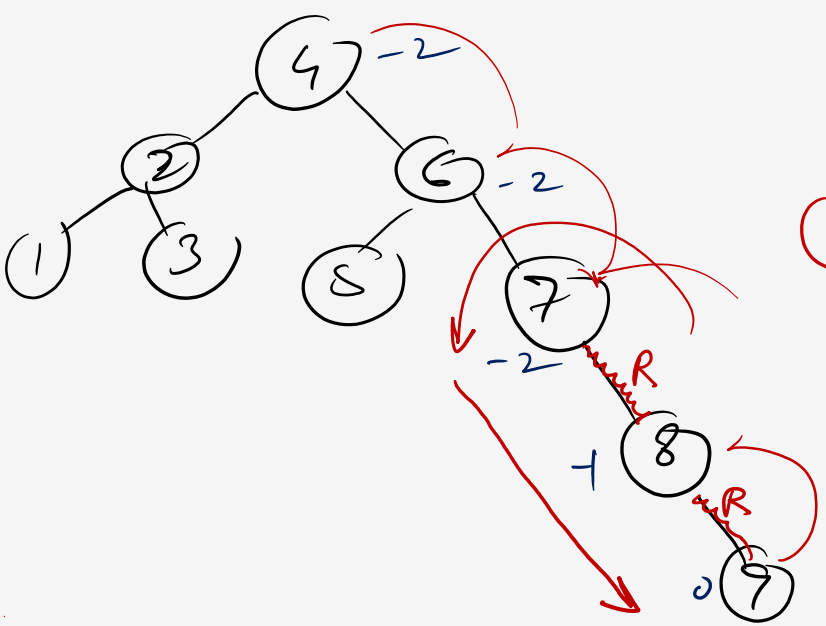
- Search ✓
- Insert ✓
- Delete ✓

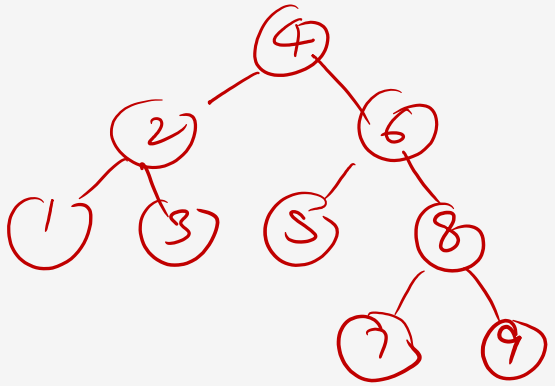
Search:- Similar to BST

TC ⇒ :

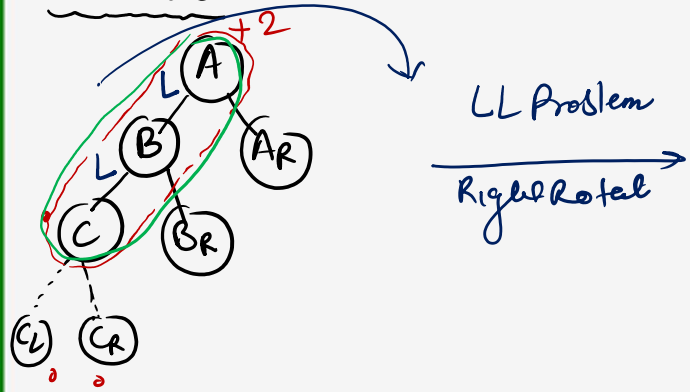
WC = O(log n)

Insert :-



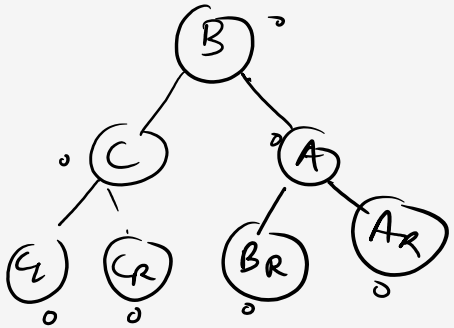


LL-Problem:

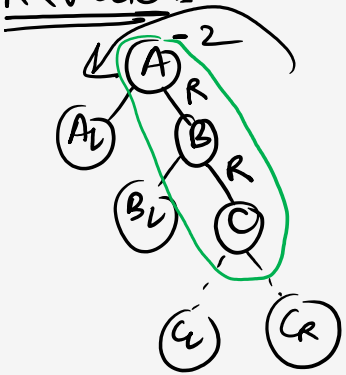


LL Problem

Right Rotate

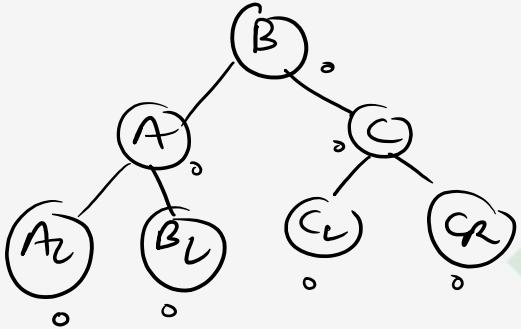


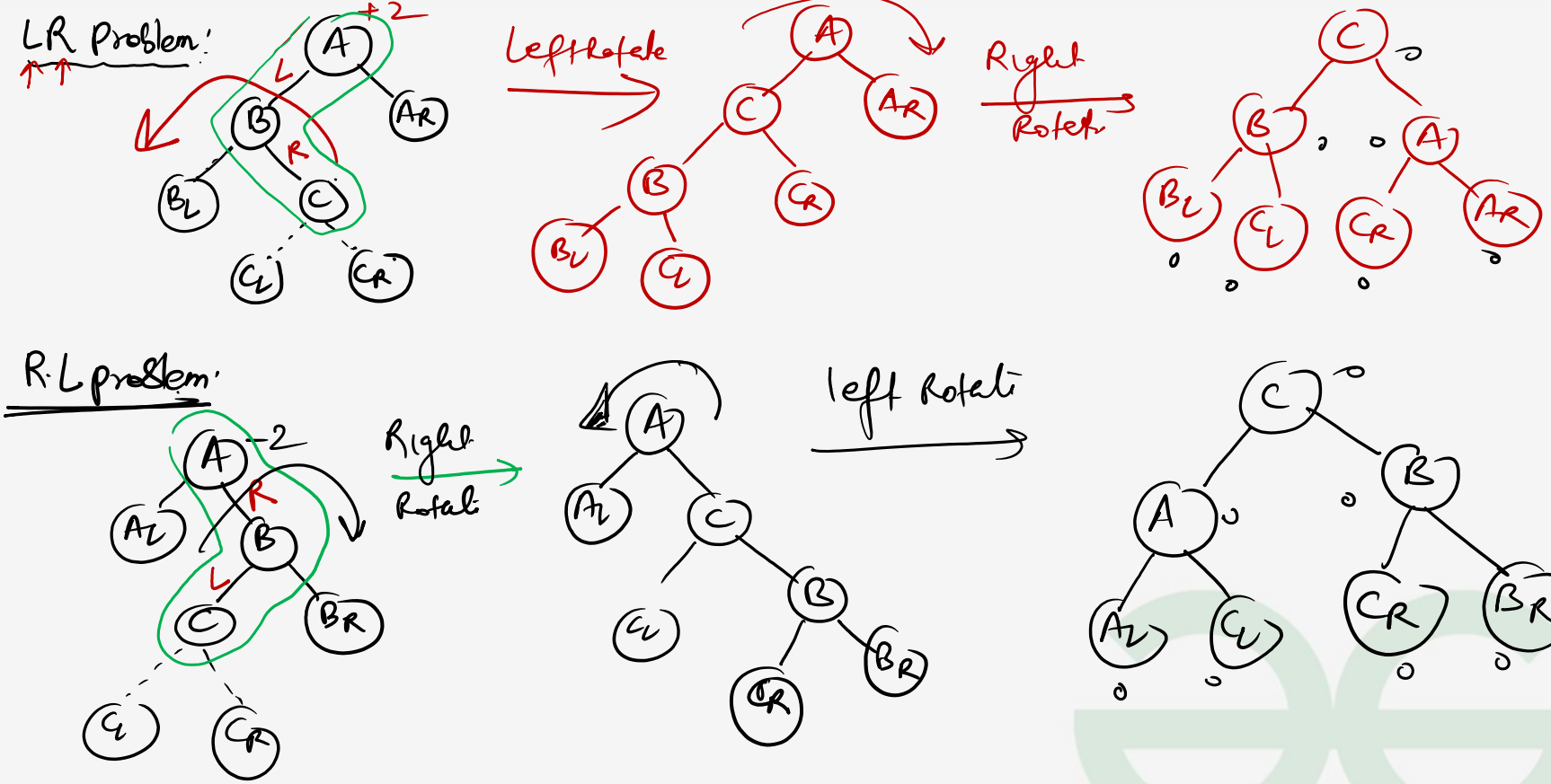
RR Problem:



RR Problem

Left Rotate




LR Problem:
+2
A
L
B
R
AR
BL
C
CL
CR
Left Rotate
A
C
AR
B
CR
BL
CL
Right Rotate
C
B
A
BL
CL
CR
AR
R.L problem:
-2
A
R
AL
B
L
C
BR
CL
CR
Right Rotate
A
AL
C
B
CL
BR
CR
left Rotate
C
B
A
AL
CL
CR
BR

Construction of AVL-tree :-

1) The creation of AVL tree is same as BST by inserting one element at a time

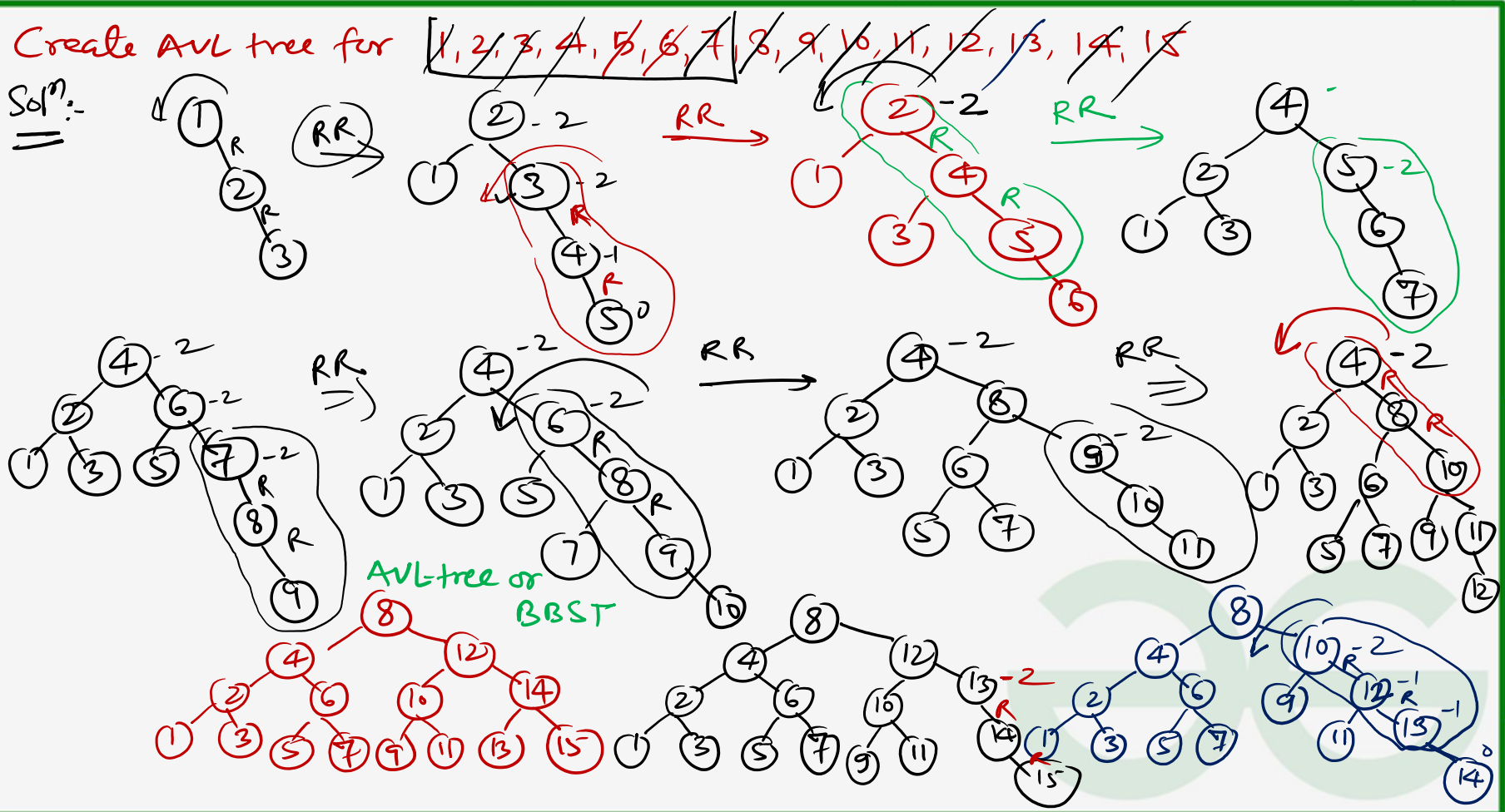
2) When the new element is inserted the balance factor of all its ancestors will be updated

3) if any node is becoming imbalance then we need to apply the proper rotation technique and make it balanced

Question: Create AVL-tree for 1, 2, 3, 4, 5, 6, 7

Create AVL tree for 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

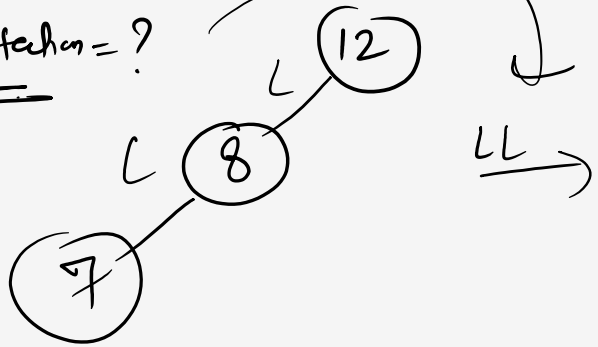
Soln:-

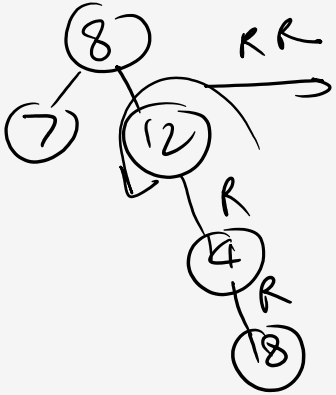


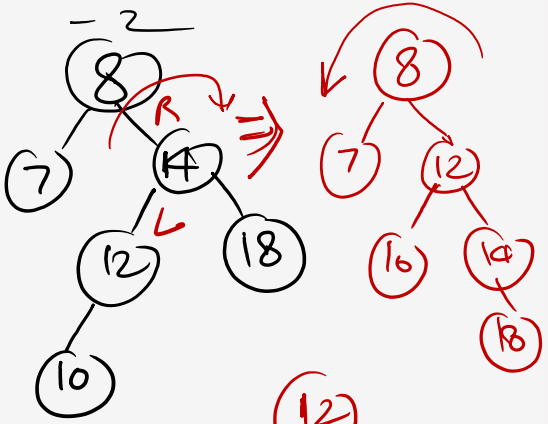
Question: Construct AVL-tree for the given keys

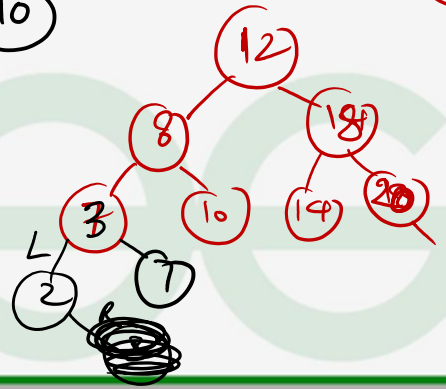
12, 8, 7, 14, 18, 16, 26, 2, 3

Total Rotation = ?





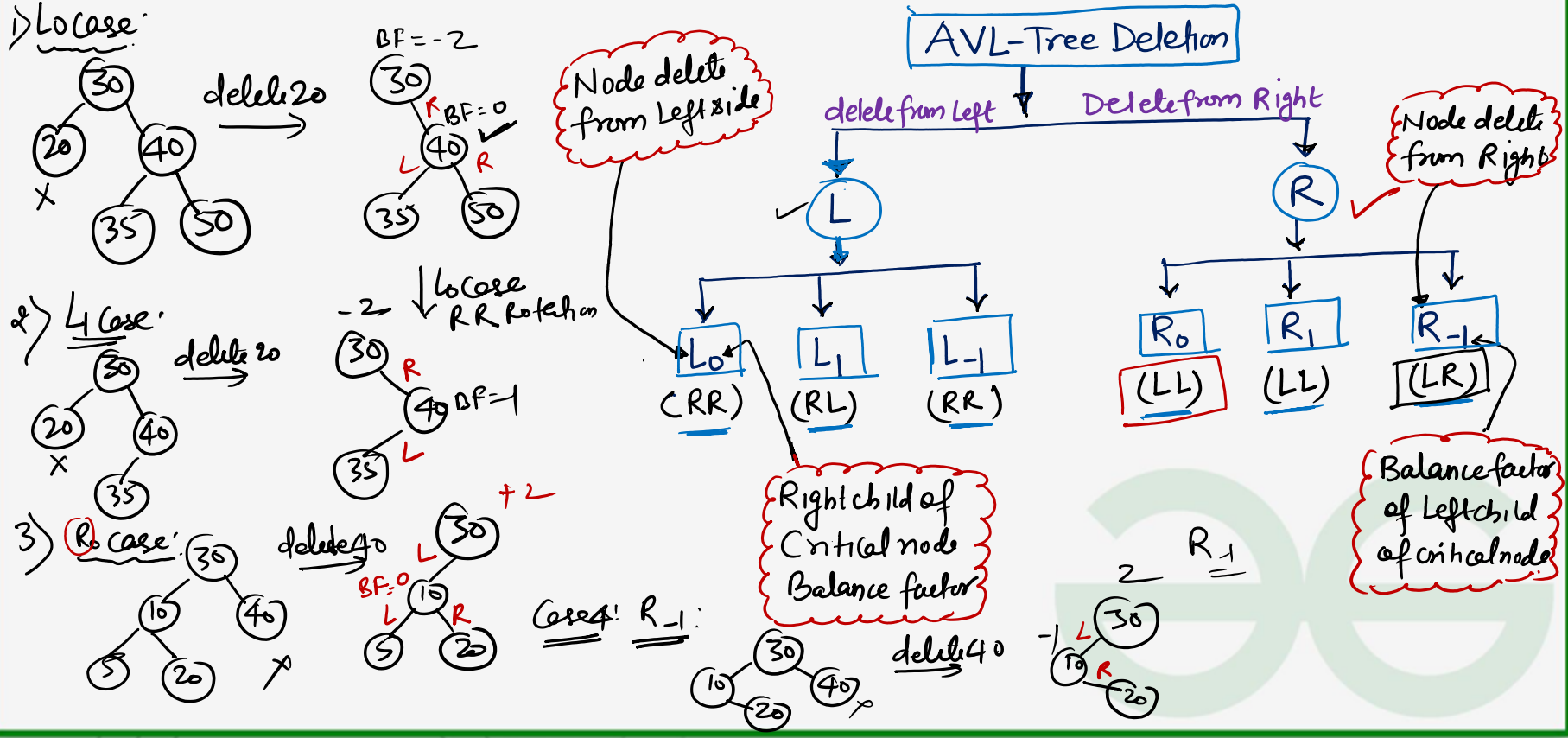




(1) + (1) + (2) + (1) + (2)

(7) Ans

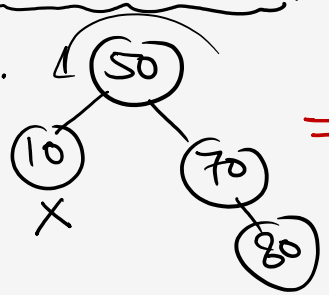
Deletion in AVL tree:-

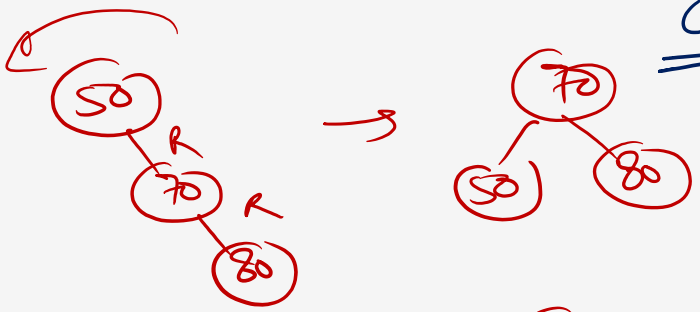


Deletion in AVL Tree:

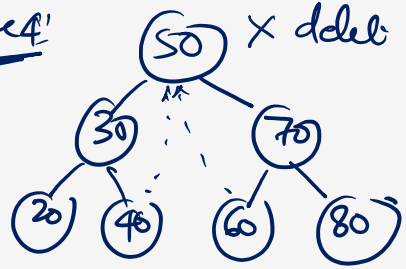
Cases:

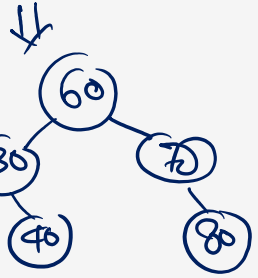
delete



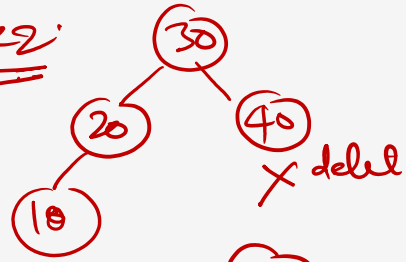


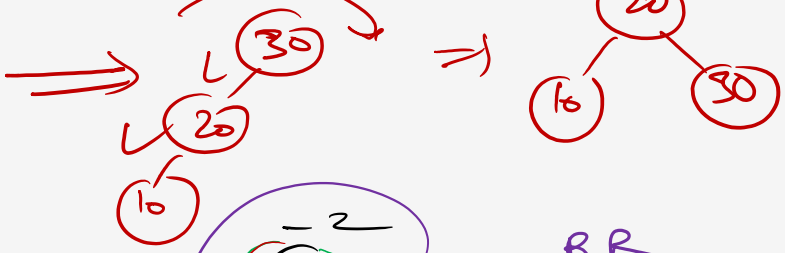
Case 4:



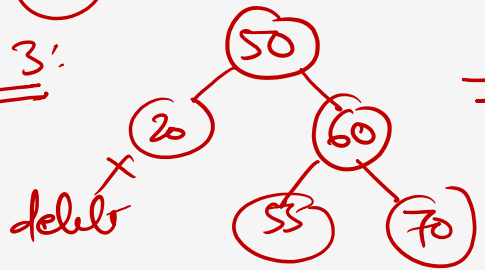


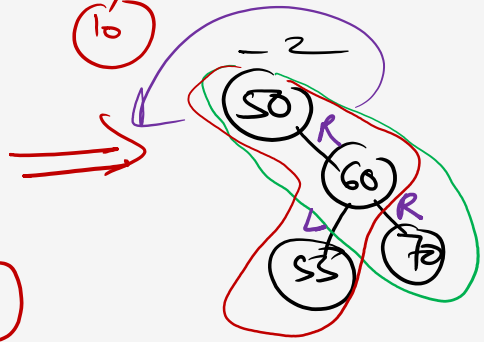
Case 2:



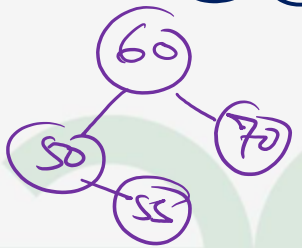


Case 3:

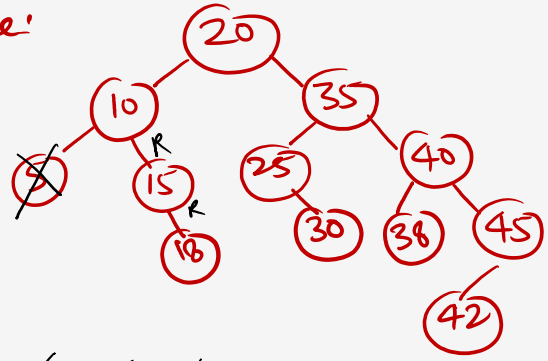


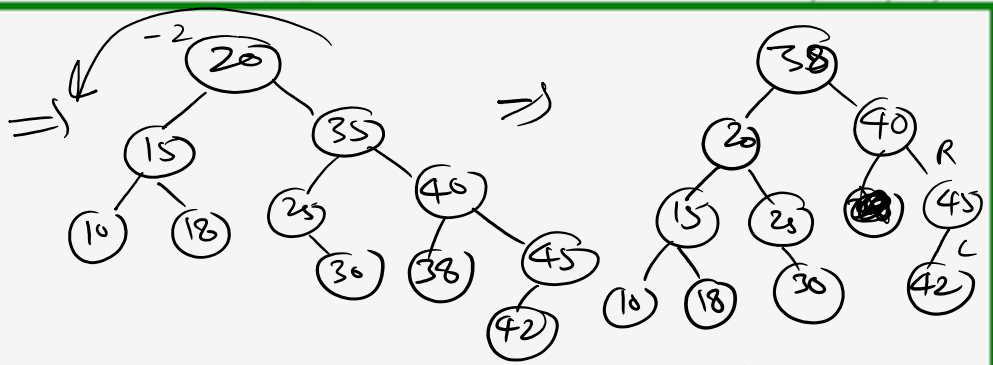


RR

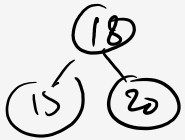


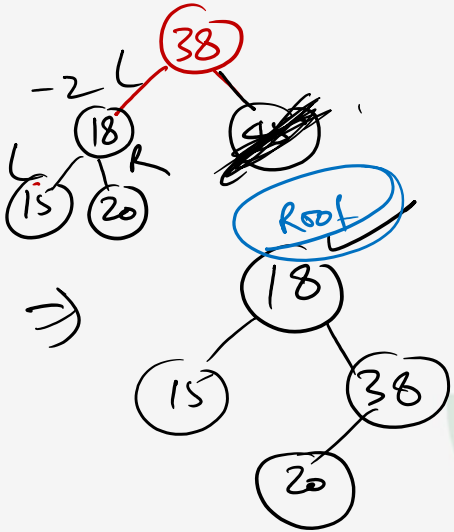
AVL-Tree:

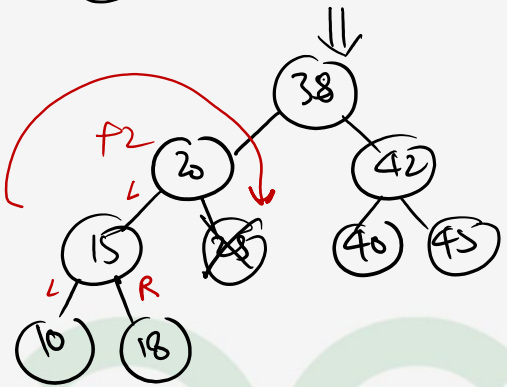




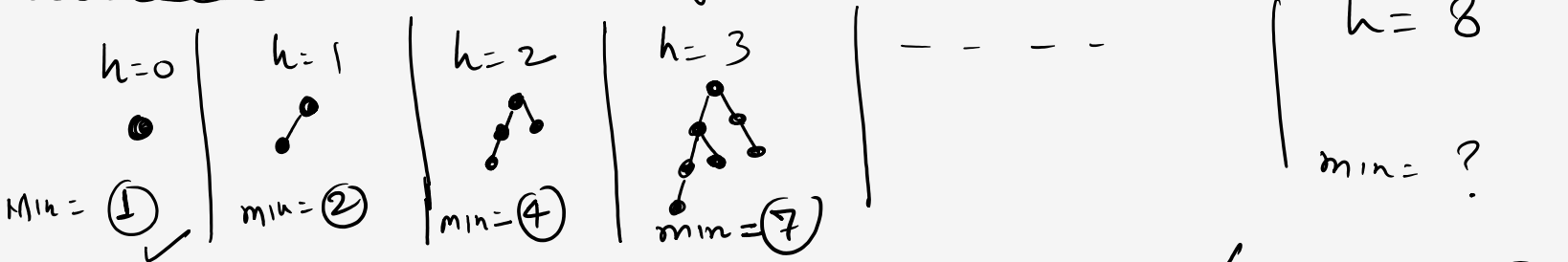
delete: 5, 35, 30, 25, 10, 40, 42, 45,







Mini no of nodes in AVL-tree on height h:



AVL:

h:	0	1	2	3	4	5	6	7	8	9	10
N(h):	1	2	4	7	12	20	33	54	88	–	–

Any

min nods = 88

maxheight = ?

AVL-Tree Property :-

1) Maximum Possible number of nodes in AVL-tree of height h = $2^{h+1} - 1$

2) Minimum no. of nodes in AVL-tree of height h is given by a recursive relation

$$N(h) = N(h-1) + N(h-2) + 1$$

Question: Suppose we have a balanced binary search tree T holding n numbers we are given two number L and H and wish to sum up all the numbers in T that lie between L and H suppose There are 'm' such number, if the tightest upperbound on the time to compute the Sum is $O(n^a \log^b n + m^c \log^d n)$ The value of $a + 10b + 100c + 1000d$?

$0 + 10 \times 1 + 100 \times 1 + 1000 \times 0 \Rightarrow 110$

Soln:

(1) L & H $\quad O(\lg n) \quad \Rightarrow 110$

(2) Traverse m elements b/w L & H $\rightarrow O(m)$

$\Rightarrow O(m + \lg n) = O(n^a \lg^b n + m^c \lg^d n)$

$a = 0, b = 1, c = 1, d = 0$

Thank You !