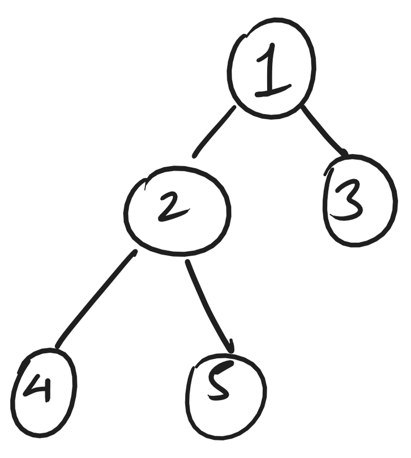
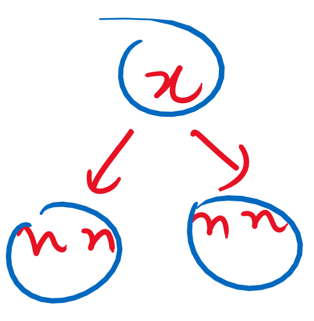
Lowest Common Ancestor (LCA) of two nodes :→ (LeetCode)

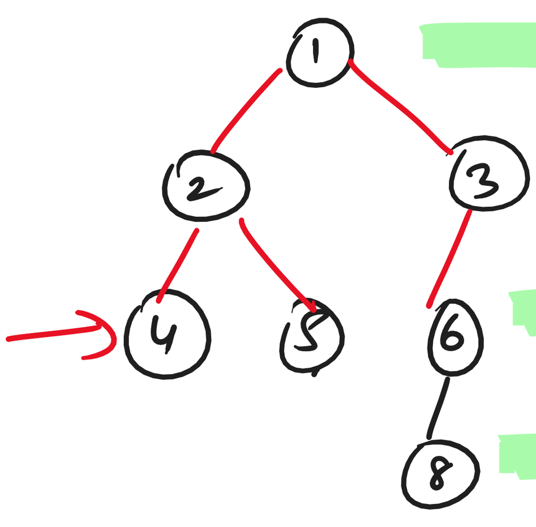


lca (root, p, q)



Pseudocode:
(i) root is null
return null
(ii) root is p or q return p or q
(iii) if l != null & r != null
return root
(iv) l != null or r != null
l r

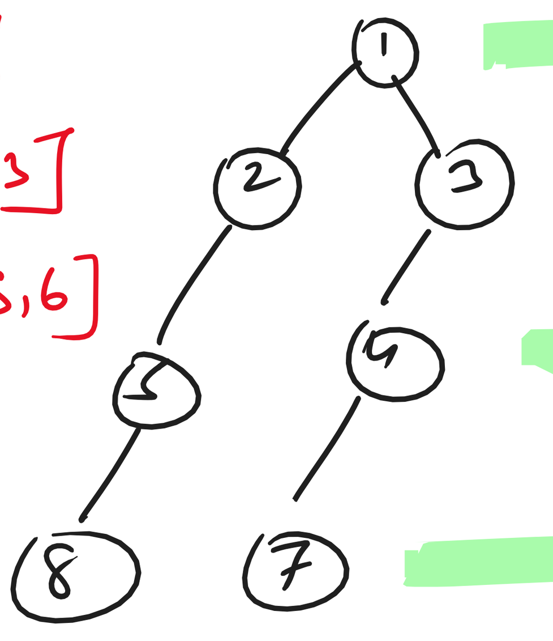
(Level Order Traversal (BFS)) :→



L0 [1]
L1 [2,3]
L2 [4,5,6]
L3 [8]

Left view 1, 2, 4, 8
i == 0 print

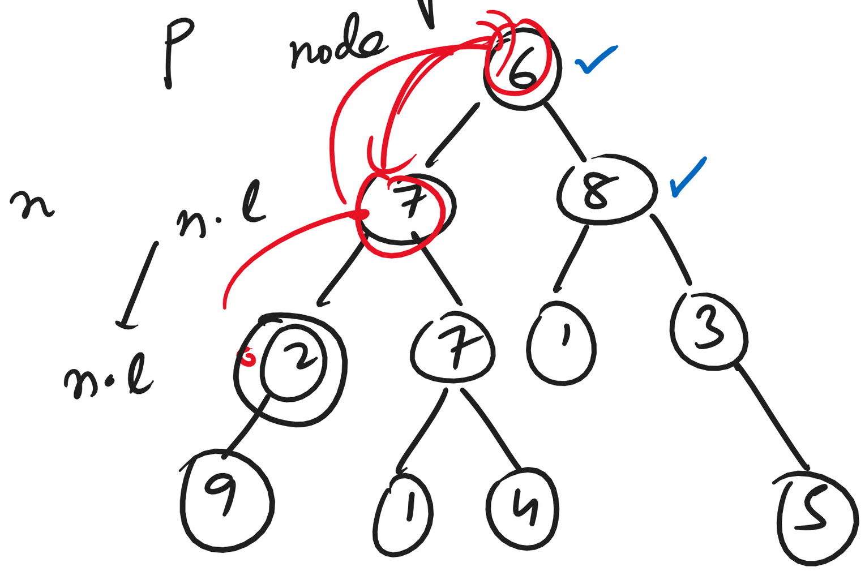
Left & Right Views



L0 [1]
L1 [2,3]
L2 [5,4]
L3 [8,7]

Right View
1, 3, 4, 7
i == size-1
print

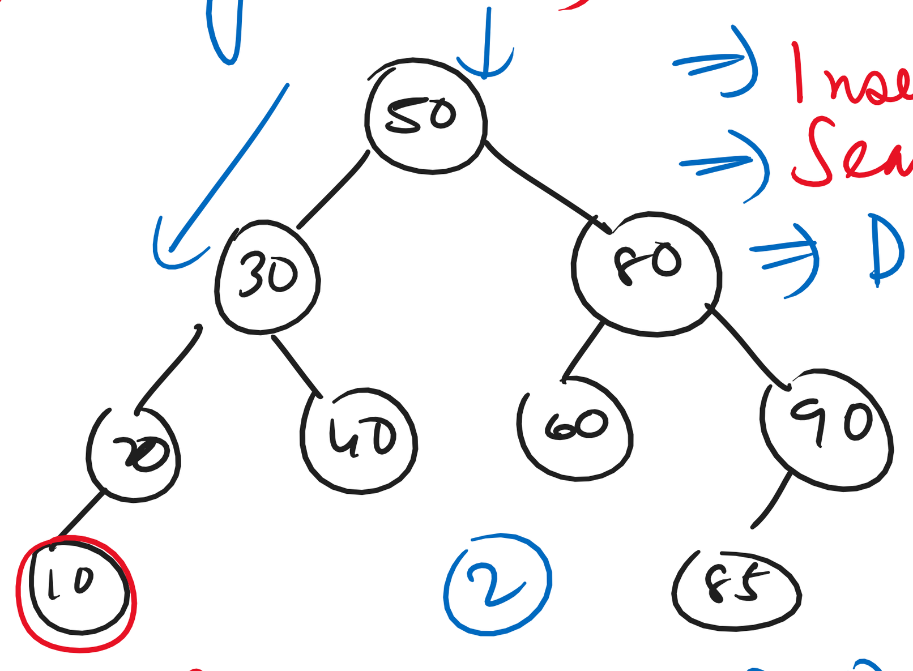
* (Even) Grand Parent Sum :→



2 7 1 3 5
(18)

(dfs)
(node, null, null, sum)
p gp
n.left, node, gp, sum
if gp != null &
gp.data % 2 == 0

(Binary Search) Tree :→



(L<N<R) Each Node
⇒ Insert → (log N)
⇒ Search → (log N)
⇒ Delete ( )

Limitation
Height is not balanced

Sorted array: ⇒ 1 2 3 4 5
⇒ 5 4 3 2 1
skewed

O(n) or O(n)

Sorted Array → BST
O(n) Linked List

Fenwick Tree
Binary Index Tree
Segment Tree
B Trees | B+ Trees
Orthogonal Tree
K-dimensional Trees
Suffix Tree

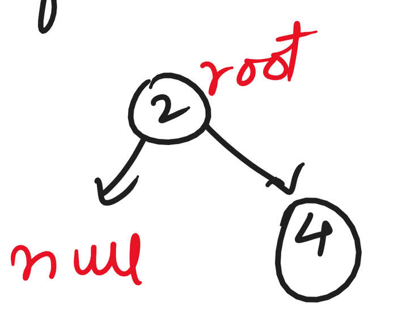
→ (Trie) ***
(LCP)

Browser auto complete
Phone Book / Contacts
Word Dictionary
Telephone Directory

* Give the three cases in the delete function in a BST :

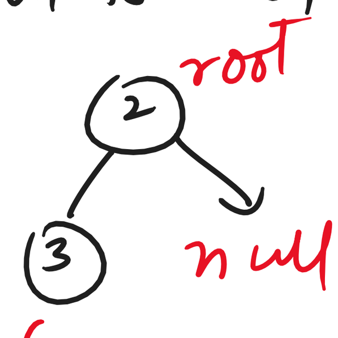
(Case I) Only One Child :→

left is null



if (r.l == null)
Temp = r.r;
delete root;
return r.r;

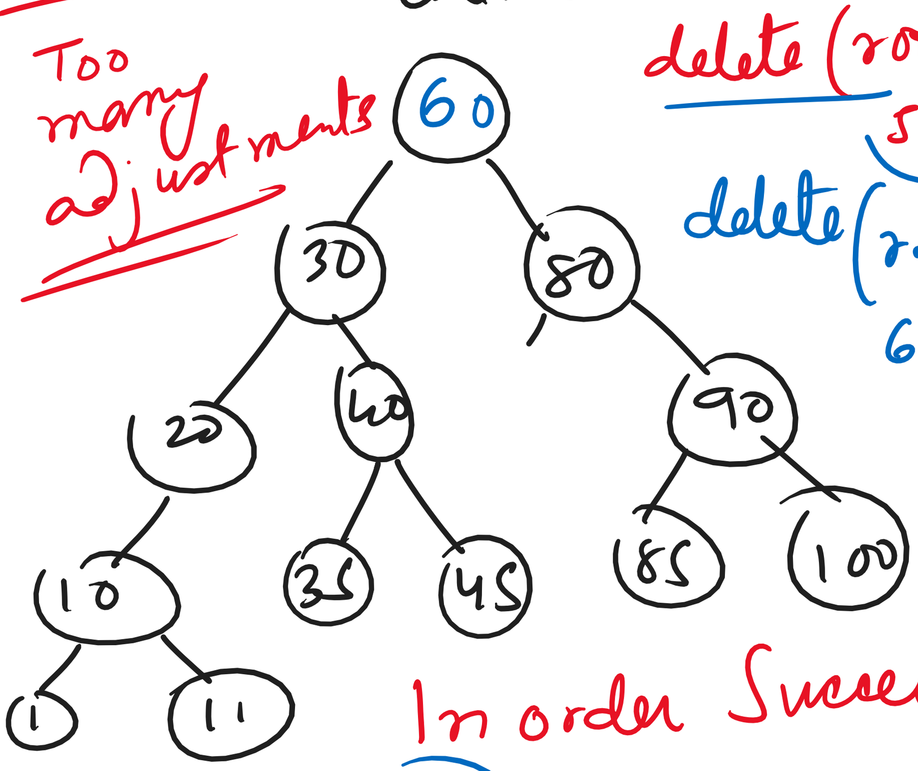
right is null



if (r.r == null)
Temp = r.l;
delete root;
return r.l;

Case II Node with two children :

Too many adjustments



delete (root, 50)
delete (r.r, 60)

Inorder Successor

Inorder: 1, 10, 11, 20, 30, 35, 40, 45, (50), (60), 80, 85, 90, 100

Inorder Successor = Least value in right subtree

BST interview question :→

Sorted Array to Balanced BST :→

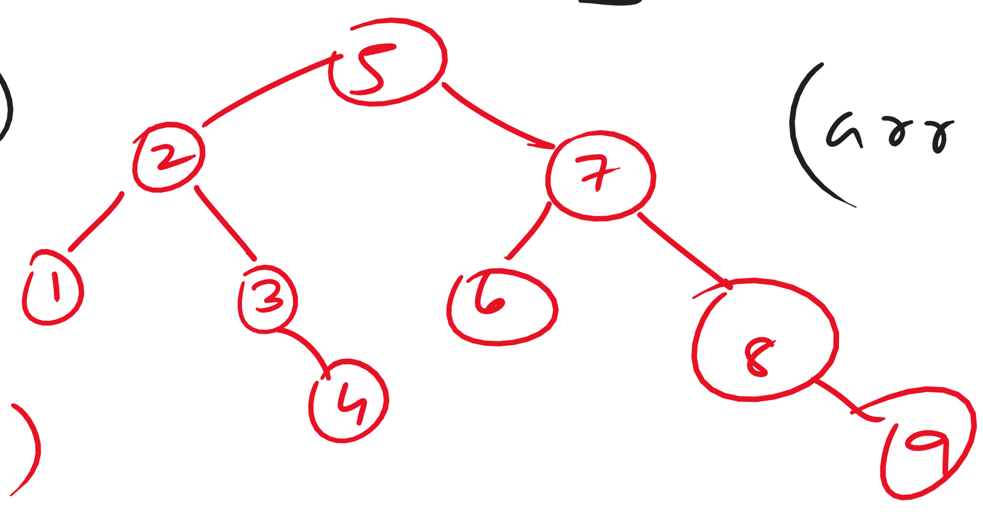
arr = {1, 2, 3, 4, 5, 6, 7, 8, 9};
index: 0 1 2 3 4 5 6 7 8
s (at 0), e (at 8)

$m = \frac{s+e}{2} = \frac{8}{2} = (4)$

$\frac{3}{2} = 1$

root = arr[mid]

(arr, s, m-1)   (arr, m+1, e)



lh - rh
3 - 4 = -1
(0 & ±1)