Binary Trees (Level Order Traversal) (Breadth First Search)

Level 0
Level 1
Level 2

At every level, we are supposed to print the data of each node from the leftmost child to the rightmost child. This should be performed from top to bottom, level by level.

* A queue is preferred for this purpose because of its FIFO property.
* Each level, all nodes are inserted into the queue in L → R sequence.
* The nodes are displayed first to rear of the queue.

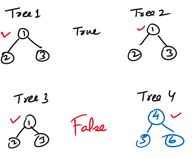
O/P: BFS → 1, 2, 3, 4, 5, 6, 7
(L0 | L1 | L2)

* Binary Trees Important Interview Questions:

TCS | Accenture | IBM | Infosys | Oracle | Capgemini | Cognizant

1. Identical Trees
2. Mirror of a Binary Tree
3. Left & Right views of a Binary Tree
4. Height of a Binary Tree
5. Symmetric Tree
6. Diameter of a Binary Tree
7. Sum of even grandparents in a binary tree.
8. Lowest Common Ancestor of two nodes
9. Serialize | Deserialize Binary Tree
10. House Robber III

Identical Trees



Conditions to be checked:

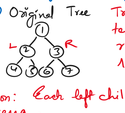
(I) Both the trees are empty: && → true

(II) One of them is null: → false ||

(III) Root of both are not same: → false ==

(IV) Left & right recursion.

* Mirror of a Binary Tree: Cognizant Sep 2024 | April 2025



Observation: Each left child has become the right child & vice versa.

In-order:
4, 2, 5, 1, 6, 3, 7

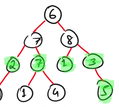
In-order:
7, 3, 6, 1, 5, 2, 4

77.7% of the time you must use recursion.

* Find the sum of the nodes in a Binary Tree whose grandparents are even.

(n.data % 2 == 0)

Sample Input: →



(sum, root, null, null)

EGS = 2 + 7 + 1 + 3 + 5 = 18.

Approach:

Sum = 0
gp.data % 2 == 0
(sum += gp.data)

Lowest Common Ancestor of two nodes in a Binary Tree

LeetCode 236 → Recursion Tree

(r, p, q) (1, 1, 3) = (I)
(r, p, q) (1, 2, 3) = (I)
(r, p, q) (1, 4, 5) = (II)

(Recursive)

LCA (1, 4, 5)
→ LCA (2, 4, 5) p, q
L→ LCA (4, 4, 5) → returns 4
R→ LCA (5, 4, 5) → returns 5
→ left = 4 & right = 5 → return (2)
→ LCA (3, 4, 5)
→ LCA (null, 4, 5) → returns null
→ LCA (null, 4, 5) → returns null
→ (2) → left = null & right = null → return null

Pseudo Code:

(I) if root is null or root is p or root is q: return root;

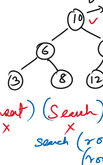
(II) if root.left != null && root.right != null: return root;

(III) if root.left == null or root.right == null: return the non-null child

* Introduction to Search Trees: → Binary Search Tree

Each node in a BST follows a very unique rule:

Left < Node < Right (L < N < R)



Height in BST is not strictly balanced.

(Insert) (Search) (Delete) (3 cases)

search (root, 18)
(root, 3)

TC (log N)

(Drawbacks)

Sorted Array:
1, 2, 3, 4, 5
O(n) O(h)
h
Right Skewed Tree
Left Skewed Tree
O(h) O(n)

(Red Black Tree) & (AVL Tree)

Height is strictly Balanced BSTs

(I) [0 to ± 1]
+2 −2
5 − 6 = −1

(m(Lh, Rh) + 1)
Lh = 5
Rh = 4
h = 6

balance factor (Lh − Rh)
5 − 4 = 1