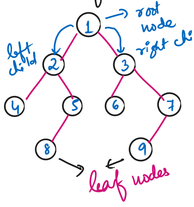
Introduction to Non-Linear Data Structures :→

① Trees :

Types: (i) Binary Tree ✓
(ii) Binary Search Tree (BST) ✓
(iii) AVL Tree
(iv) Red Black Tree
(v) Complete Binary Tree → (Heaps) ✓
(vi) Binary Indexed Tree (Fenwick Tree)
(vii) Segment Tree
(viii) Orthogonal Range Trees
(ix) K-dimensional Trees
(x) Skewed Trees ✓
(xi) Suffix Trees (Tries) ✓

Binary Tree : It's a non-linear data structure where there are entities called nodes. Each node has children nodes called left or right child. The starting node of the tree is called the root node.
Note: If a node has no children, it is called a leaf node.



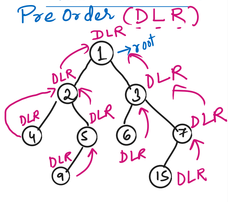
Level 0 / Level 1 / Level 2 / Level 3
or
Level 1 / Level 2 / Level 3 / Level 4

The nodes are placed in levels :
Level (0 – n-1)
Level (1 – n)
Both are correct

Formula for non-leaf nodes : $\left(\frac{n}{2}+1\right)$ Leaf = $(n - nl)$

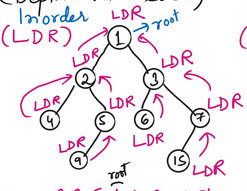
Tree Traversals:

Pre Order (DLR) (DFS) (Recursion) (Depth First Search)



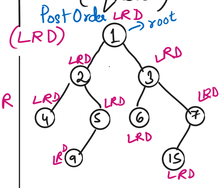
O/p: 1,2,4,5,9,3,6,7,15 (root first)

Inorder (LDR)



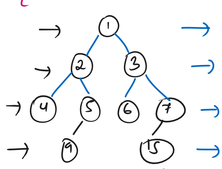
op: 4,2,9,5,1,6,3,15,7 (root = middle)

Post Order (LRD) (BFS) (queue)



O/p: 4,9,5,2,6,15,7,3,1 (root last)
(DSP / SNS)

{Breadth First Search} or {Level Order Traversal}

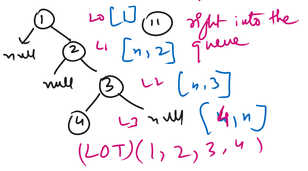


L0 → L1 → L2 → L3

O/p: Top to Bottom, Level By Level, Left to Right
⇒ 1, 2, 3, 4, 5, 6, 7, 9, 15

(FIFO) Queue
* For every node, we push the :
① left into the queue
② right into the queue

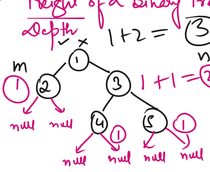
L0 queue [1] → (size)
L1 queue [2,3] →
L2 queue [4,5,6,7] →
L3 queue [9,15] →



Binary Trees Important Interview Questions :→ Leet code, Coding Ninjas, Code Chef

(i) Height of a Binary Tree
(ii) Identical Trees
(iii) Mirror of a Binary Tree
(iv) Symmetric Tree
(v) Sum of all nodes with "Even Grandparents"
(vi) Seriolize / Deserialize Binary Tree
(vii) Lowest Common Ancestor of Binary Tree (LCA)
(viii) House Robber III
(ix) Left View of Binary Tree
(x) Right View of Binary Tree

* **Height of a Binary Tree :→** (Depth) Def": The maximum no. of nodes from the root node to any of it's descendants.



Pseudo-code : left height, right height
total height = max (lh, rh) + 1