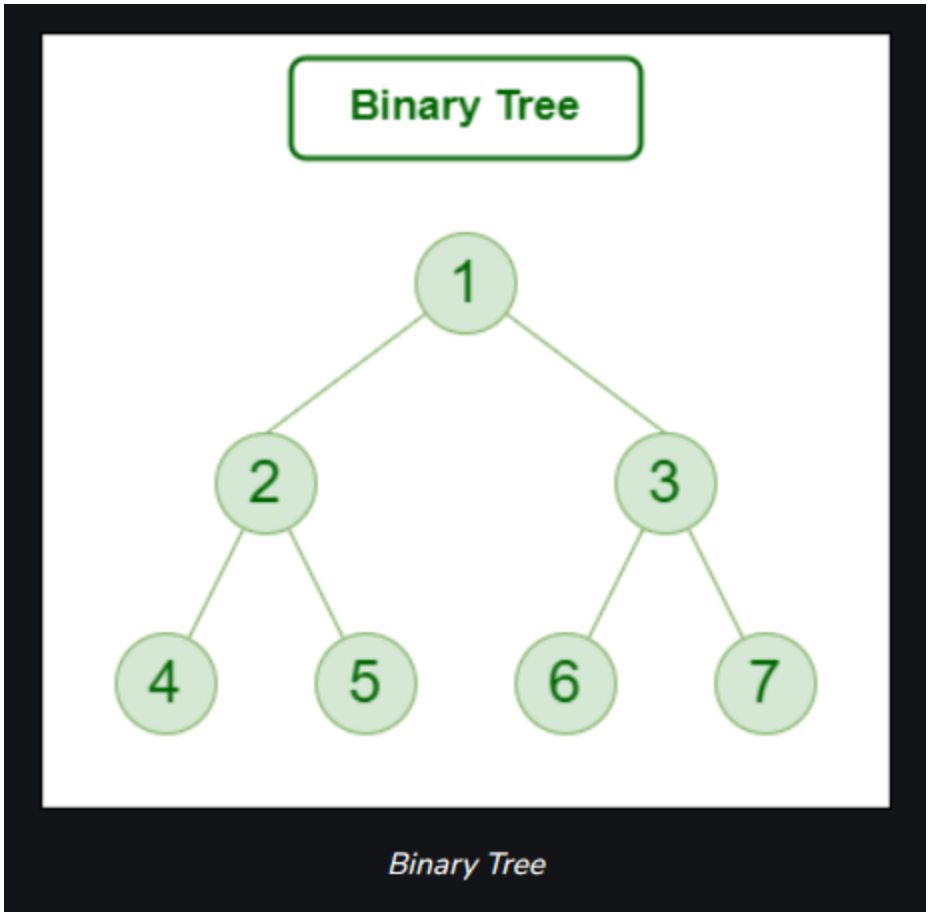
Binary Tree

****BST and Binary Tree are similar yet different structures.

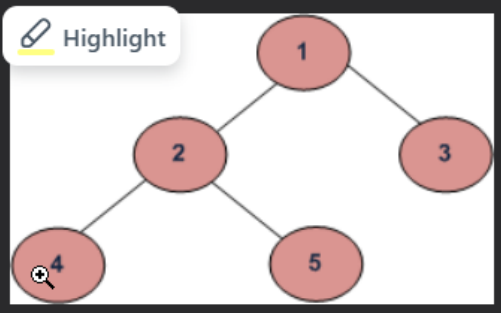


BASIC OPERATIONS	Time	Space	Exception
Insert	O(log n) [Best], O(n) [W]	O(log n) [Best], O(n) [W]	May exceed space if the tree is unbalanced
Remove	O(log n) [Best], O(n) [W]	O(log n) [Best], O(n) [W]	May exceed space if the tree is unbalanced
Search	O(log n) [Best], O(n) [W]		O(n) in degenerated (one sided) trees
Delete	O(log n) [Best], O(n) [W]		

Traversals			
Pre-order	O(n)	O(h)	h is the height of the tree
In order	O(n)	O(h)	h is the height of the tree
Post order	O(n)	O(h)	h is the height of the tree
Level order (BFS) - Naive	O(n)	O(n)	
AXILLARY OPERATIONS			
Find level	O(n)		
Find depth	O(n)	O(h)	If the tree is empty then return 0
Find size of entire tree	O(n)	O(h)	h is the height of the tree

Depth or Height of BT

Consider the following tree:



Example of Tree

maxDepth('1') = max(maxDepth('2'), maxDepth('3')) + 1 = 2 + 1

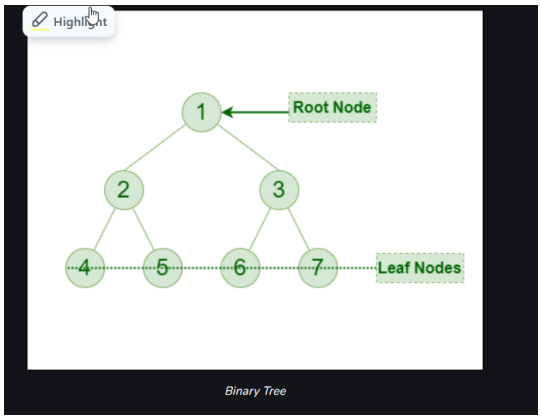
because recursively

maxDepth('2') = max (maxDepth('4'), maxDepth('5')) + 1 = 1 + 1 and (as height of both '4' and '5' are 1)

maxDepth('3') = 1

Type of Traversals

1. Depth- First search
    a. Preorder (Current, L , R)
    b. In order (L, C , R)
    c. Post order (L, R. C)
2. Breath- First search
    a. Level order traversal



Pre-order Traversal of the above tree: 1-2-4-5-3-6-7
In-order Traversal of the above tree: 4-2-5-1-6-3-7
Post-order Traversal of the above tree: 4-5-2-6-7-3-1
Level-order Traversal of the above tree: 1-2-3-4-5-6-7

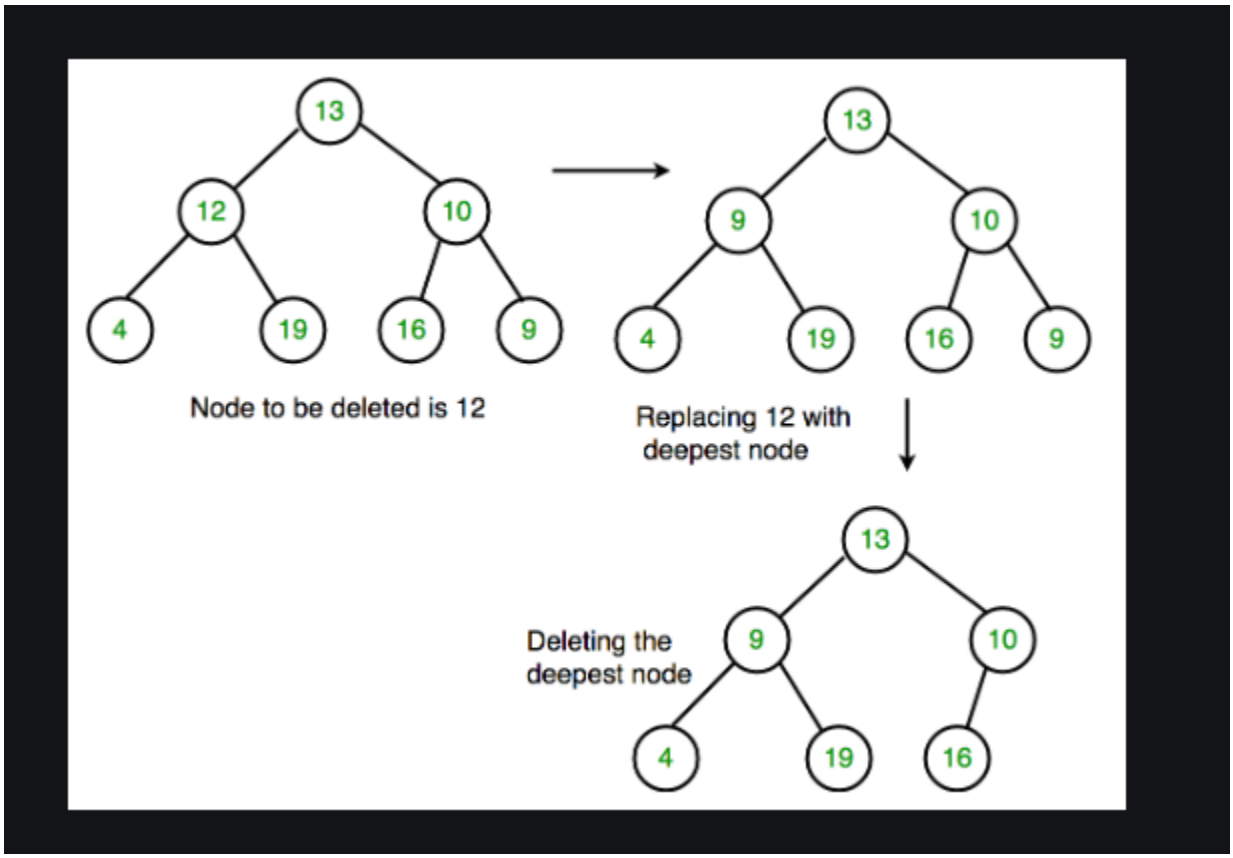
Deletion

Algorithm:

Starting at the root, find the deepest and rightmost node in the binary tree and the node which we want to delete.

Replace the deepest rightmost node's data with the node to be deleted.

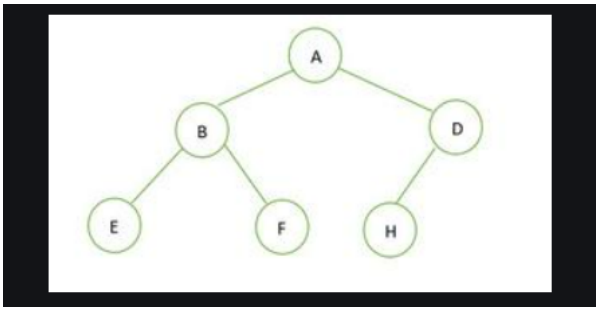
Then delete the deepest rightmost node.



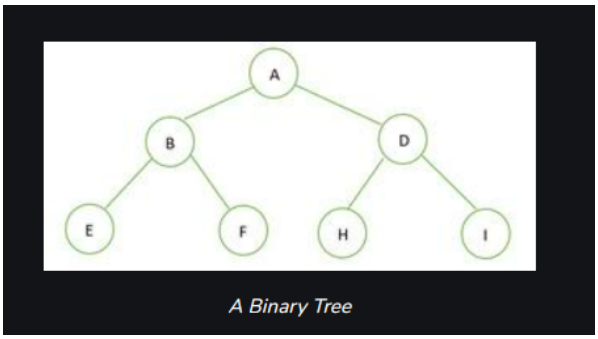
- Max number of node at the level = 2^n (where n = 0, 1, 2, 3)
- Max num of node at the whole tree = 2^n -1 (where n = 1,2,3)
- In a Binary Tree with N nodes, the minimum possible height or the minimum number of levels is Log2(N+1):
- In a balanced binary tree, the height of the left and right subtrees of every node differ by at most 1:
- Nodes having same parents are called siblings
- Internal / External : Leaf nodes are external, non- left node = internal
-

Types of Binary Tree

1. Complete Binary Tree
2. Perfect Binary Tree

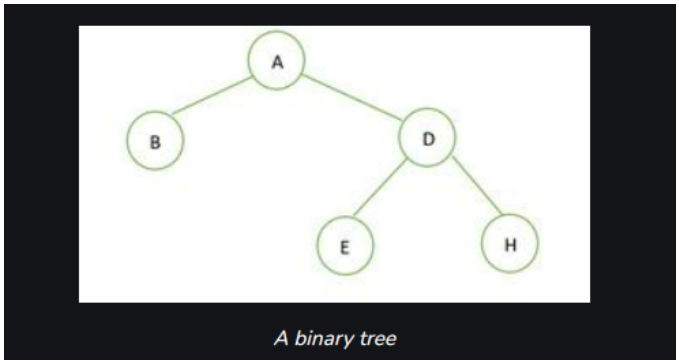


Complete tree



Perfect Tree

3. Full binary Tree



- **Real world application**
  - Excel and spread sheets
  - Decision Trees
  - Huffman encoding
  - Priority Queue
  - Game AI
  - Searching
  - Sorting : Binary tree sort, Heap sort
  -

- **Why use BINARY TREE?**
  - less memory
  - less time