Unit - 3 Non-Linear Data Structure

* Tree

→ Tree is a finite set of one or more nodes such that

① there is specially designed node called root node.

② Remaining node are partitioned into disjoint set $T_1, T_2, T_3, \ldots T_n$ where $T_1, T_2, \ldots T_n$ are called sub-tree of root

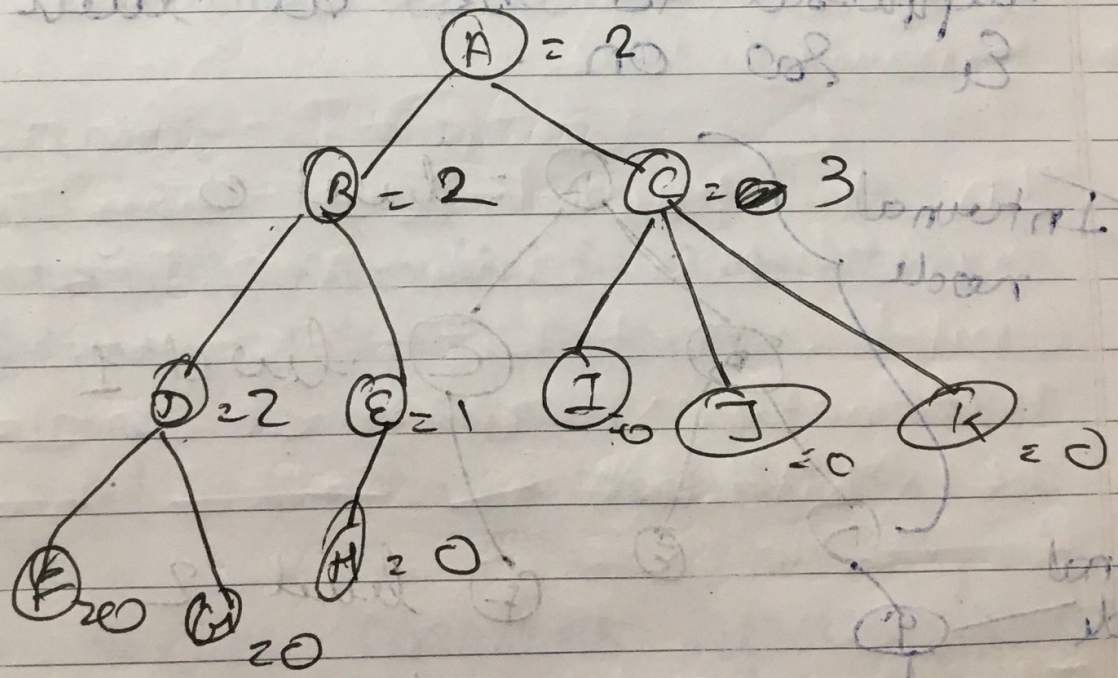
→ Root Node - A root node is unique node in tree to which further sub-tree are attached.

→ Leaves :- This are the terminal nodes of the tree.

→ Parent Node :- the node having further sub-branches is called parent node

→ Degree Of a Node (Vertices)

The total no. of sub-tree attached to that node is called Degree of node



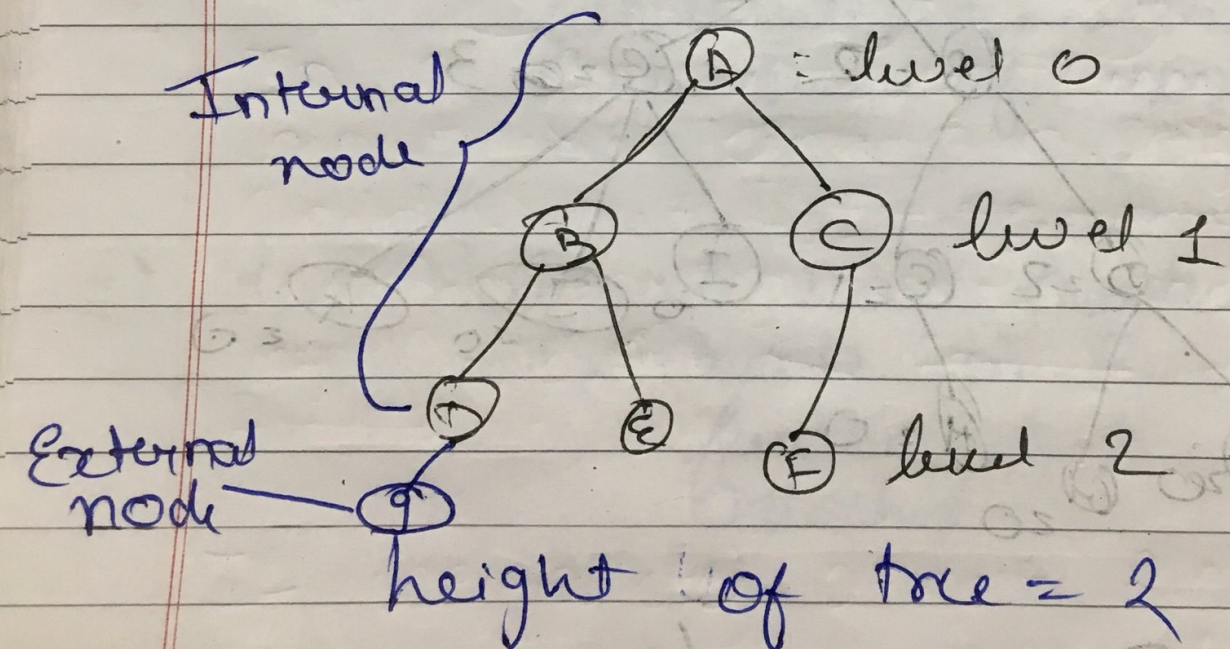
Degree of tree = 3 (max)

→ Degree Of tree

the maximum degree is the tree is degree of tree.

→ level Of tree

The root node is always consider at level 0. The adjacent to root node are supposed to be at level 1 & so on.



F has predecessor as A & C

B has successor D & G & E

→ Height Of a tree

Max. level is height of tree. It is also known as dept of tree.

→ Predecessor Or Ancestor

While displaying the tree If some particular node occurs previous to some particular node than that node is called predecessor Or Ancestor of other node.

→ Successor

It is a node which occurs next to some other node.

→ Internal & External Node

leaf node means node having no child nodes as leaf node are not having further links we called leaf node as External

node & non leaf nodes are called internal node.

→ Sibbling

Node with common parents are called Sibling

→ Forest

It is collection of disjoint tree.

→ Binary Tree

In a genral tree any node can have any no. of child. but, if we put some restrictions on the no. of child ~~it is called Binary tree~~.

If we allote atmost 2 child node, then that type of tree is call Binary tree

→ Full Binary Tree (Strictly Binary Tree)

A full Binary tree is tree in which every node has 2 or 0 children.

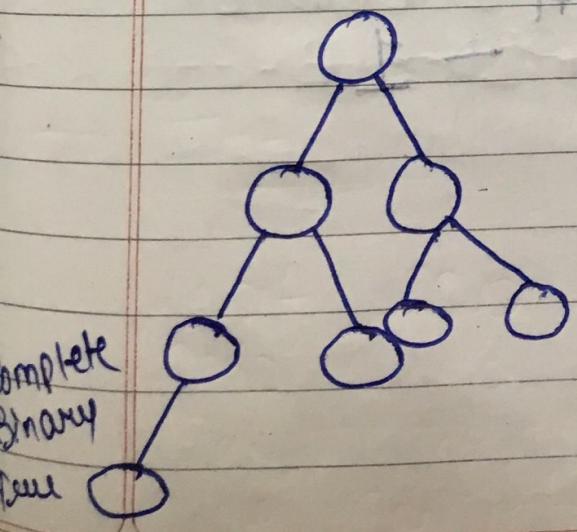
No. of leaf node = no. of internal node + 1

18/9/19

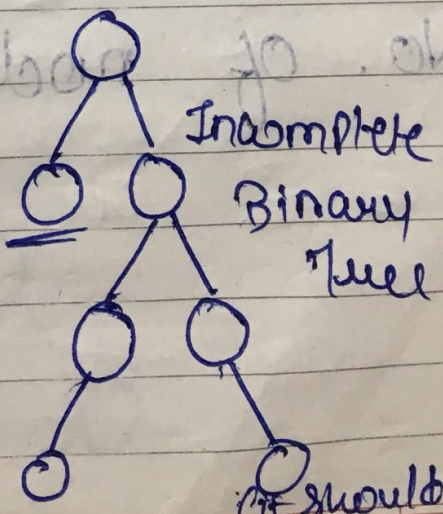
∴ leaf node(l) = I + 1

→ Complete Binary Tree

A Complete Binary Tree is Complete Binary " if all levels are Completely fill except possible the last level & last level has all keys are as left as possible.



omplete
Binary
Tree

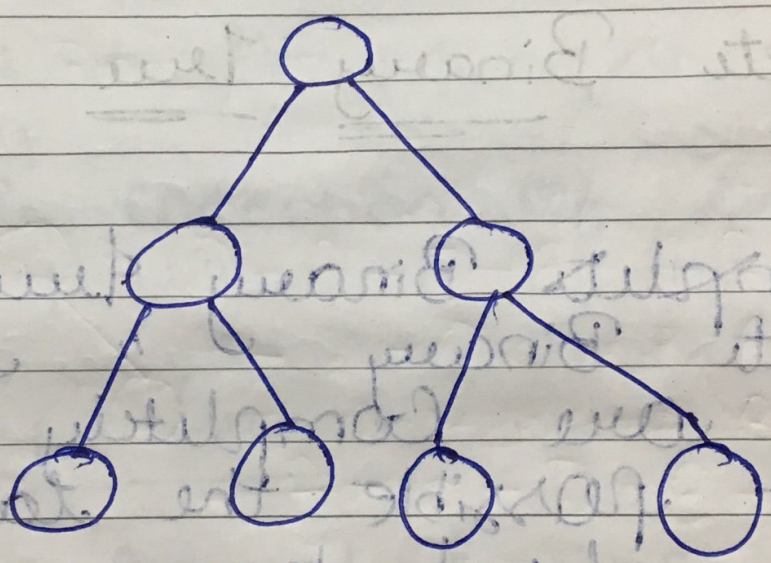


Incomplete
Binary
Tree

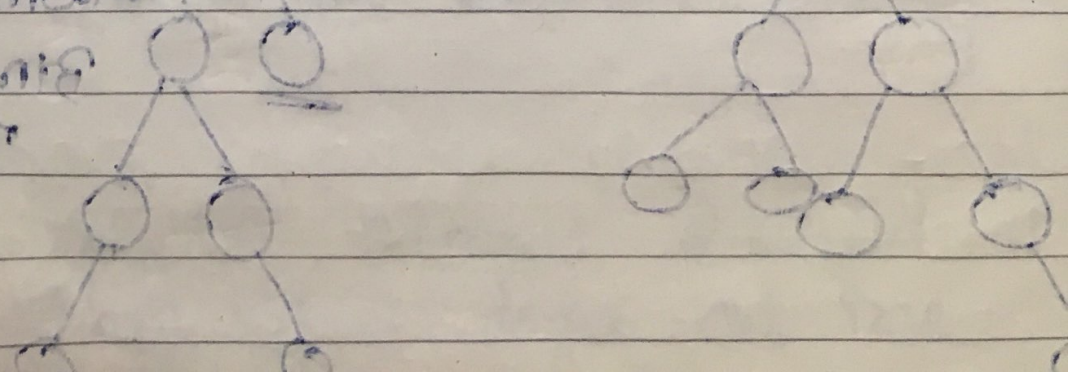
(It should be as left as poss.)

→ Perfect Binary Tree

A Binary tree is perfect Binary tree is which all internal node have 2 children & all leaf node are at the same level.

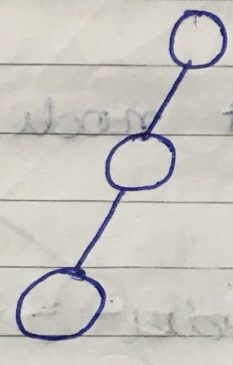


No. of node = $2^{h+1} - 1$



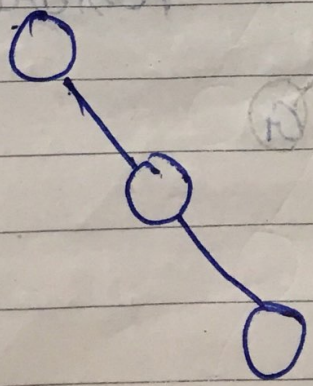
★ Left Heavy Tree

→ The tree in which each node is attached as a left child of parent node, then it is called Left Heavy Tree



★ Right - Heavy Tree

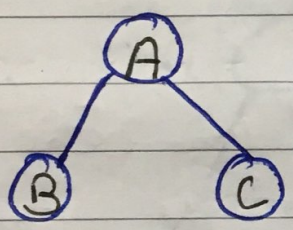
→ The tree in which each node is attached as right child of Parent node.



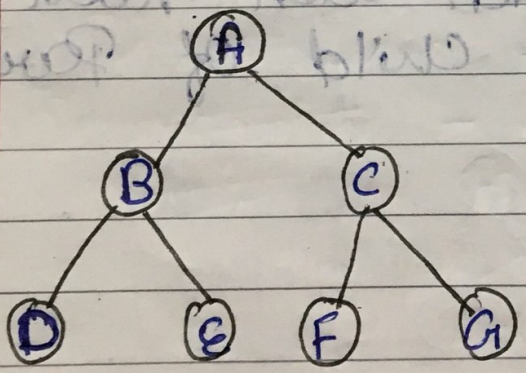
☆ Binary Tree Traversal

1. Inorder " [L D R]
2. Preorder " [D L R]
3. Postorder " [L R D]

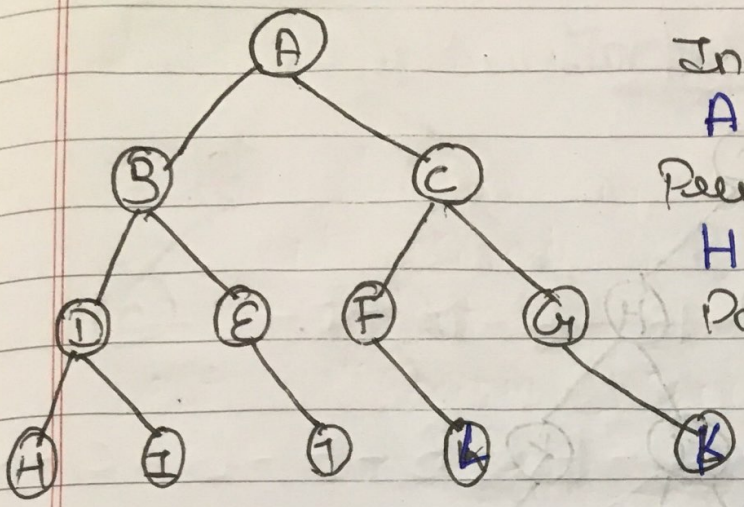
L - left
R - Right
D - Root / Parent node



Inorder = B - A - C
Preorder = A - B - C
Postorder = B - C - A



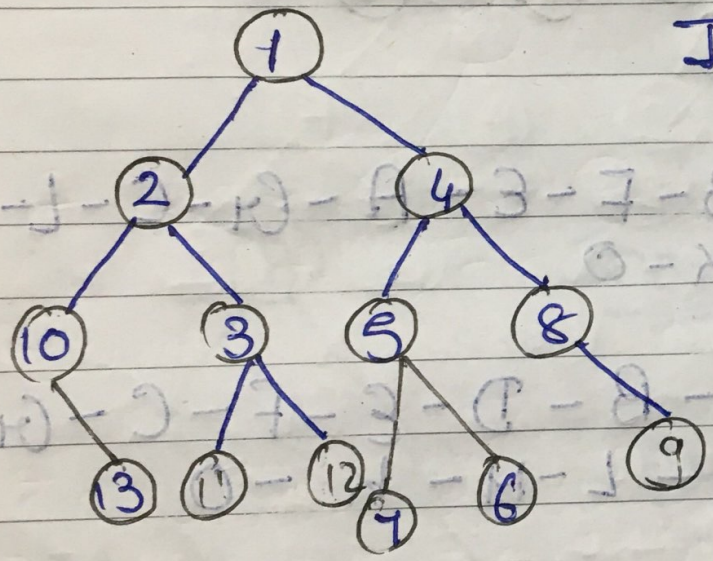
Inorder = D - B - E - A - F - C - G
Preorder = A - B - D - E - C - F - G
Postorder = D - E - B - F - G - C - A



Inorder = H-D-I-B-E-J-A-F-L-C-G-K

Preorder = A-B-D-H-I-E-J-C-F-L-G-K

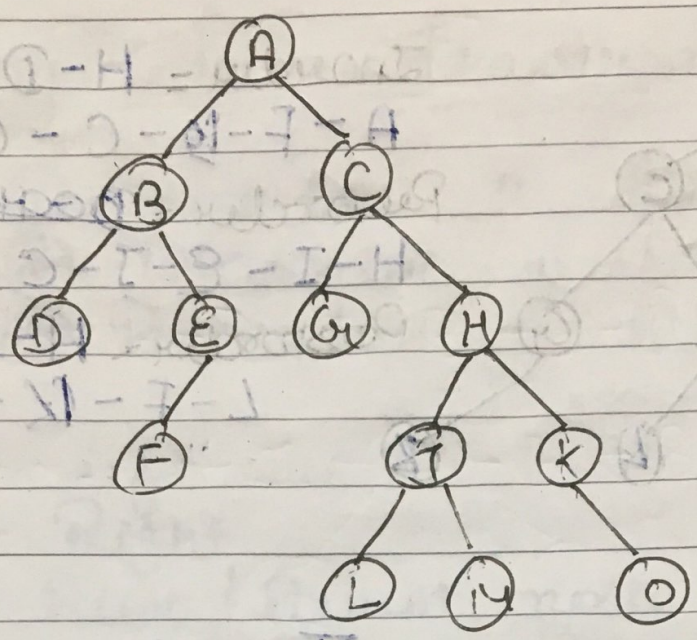
Postorder = H-I-D-J-E-B-L-F-K-G-C-A



Inorder → 10-13-2-3-11-12-1-7-5-6-4-8-9

Preorder → 1-2-10-13-3-11-12-4-5-7-6-8-9

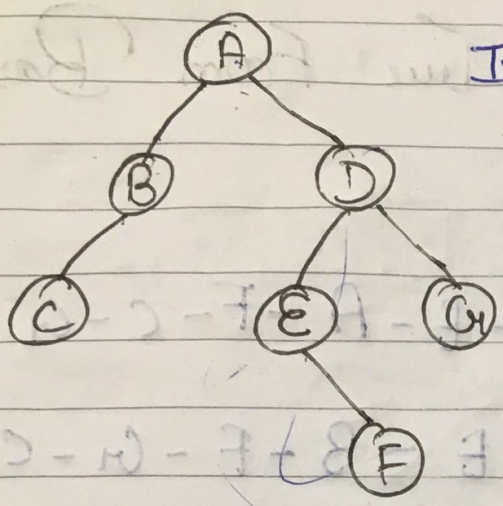
Postorder - 13-10-11-12-3-2-7-6-5-9-8-4-1



Inorder - D - B - F - E - A - G - C - L - J - M - H - K - O

Preorder - A - B - D - E - F - C - G - H - J - L - M - K - O

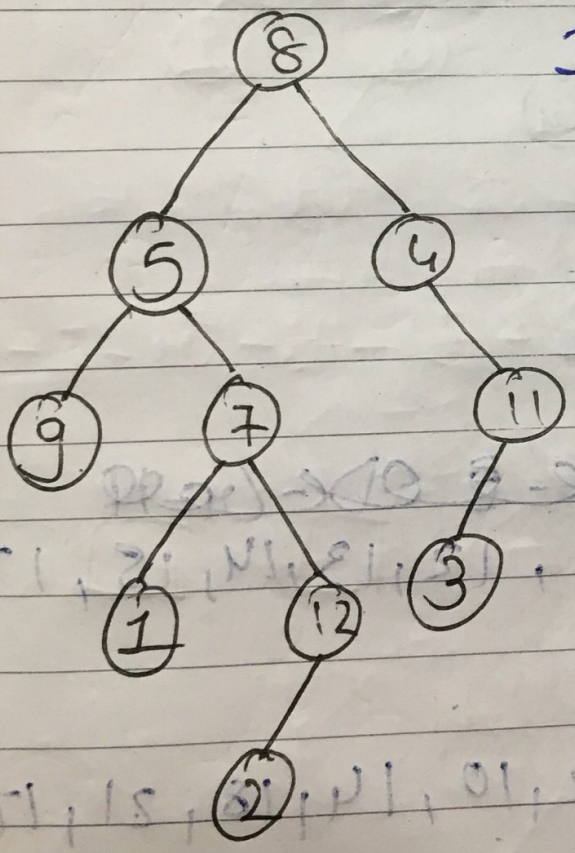
Postorder - D - F - E - B - G - L - M - J - O - K - H - C - A



Inorder → C - B - A - E - f - D - G

Preorder → A - B - C - D - E - f - G

Postorder → C - B - F - E - G - D - A



Inorder → 9 - 5 - 1 - 7 - 2 - 12 - 8 - 4 - 3 - 11

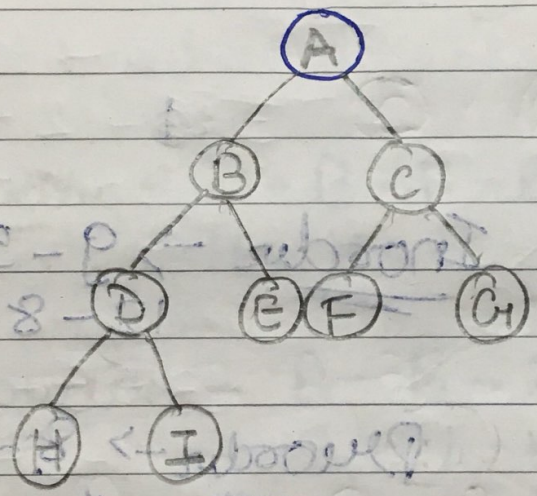
Preorder → 8 - 5 - 9 - 7 - 1 - 12 - 2 - 4 - 11 - 3

Postorder → 9 - 1 - 2 - 12 - 7 - 5 - 3 - 11 - 4 - 8

* Creation Of Binary Tree From Basic Traversal

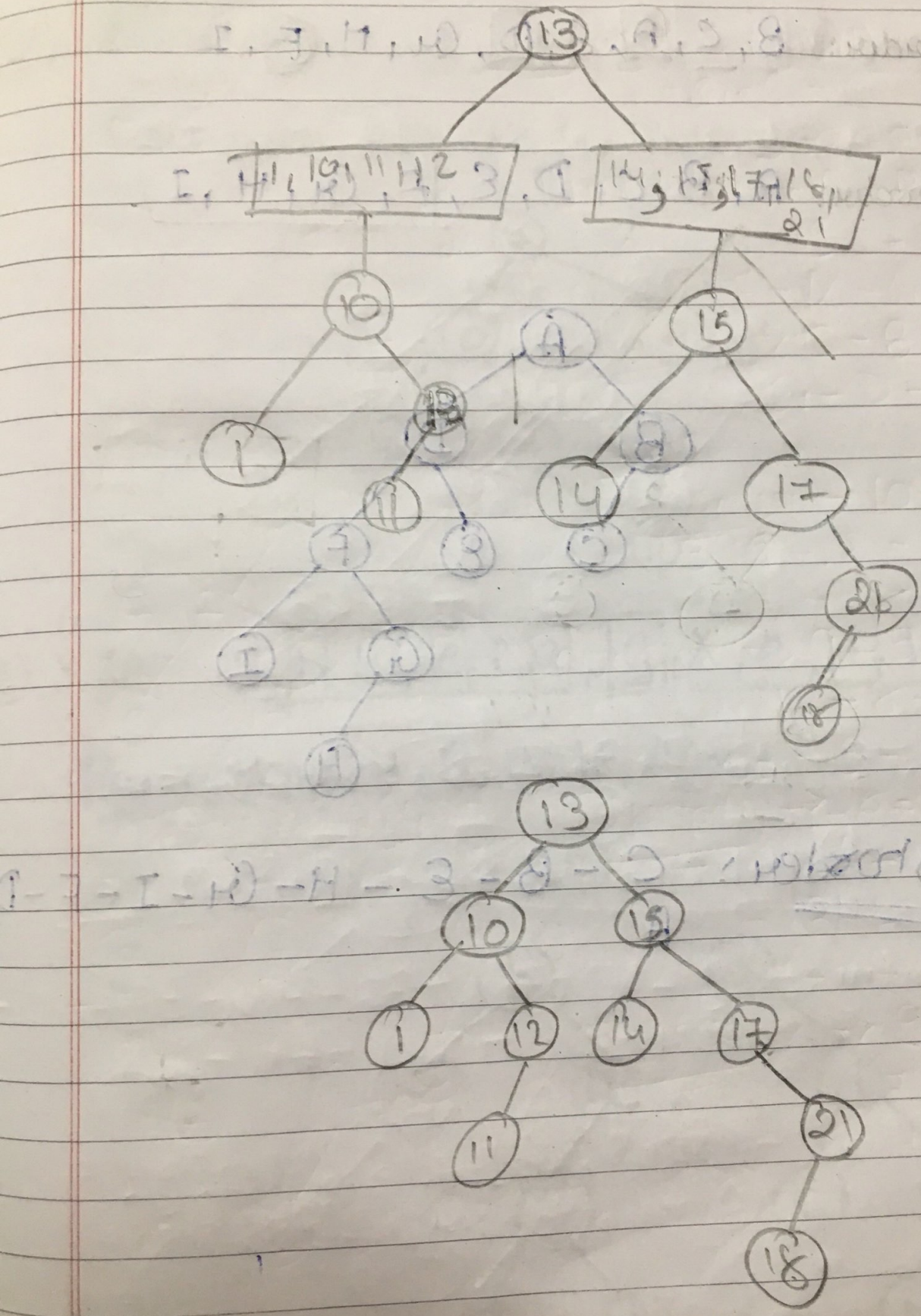
Ex: Inorder - H-D-I-B-E-A-F-C-G

Postorder - H-I-D-E-B-F-G-C-A



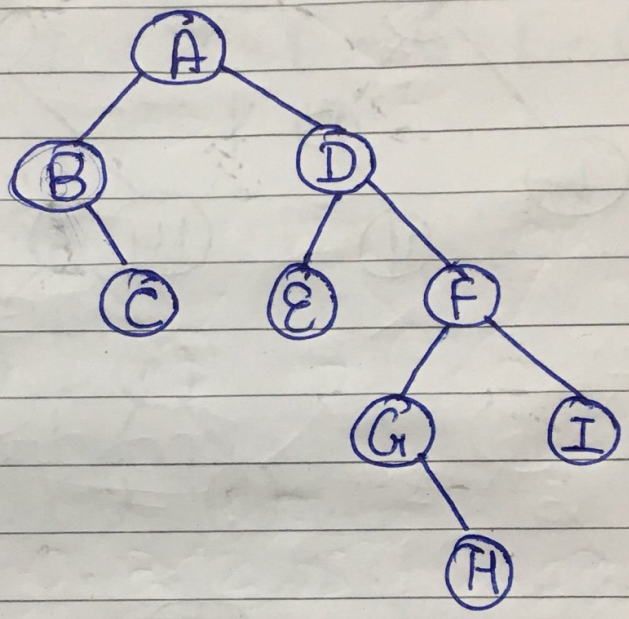
Ex: Inorder :- 1, 10, 11, 12, 13, 14, 15, 17, 18, 21

Postorder :- 1, 11, 12, 10, 14, 18, 21, 17, 15, 13



Ex: Inorder: B, C, A, E, D, G, H, F, I

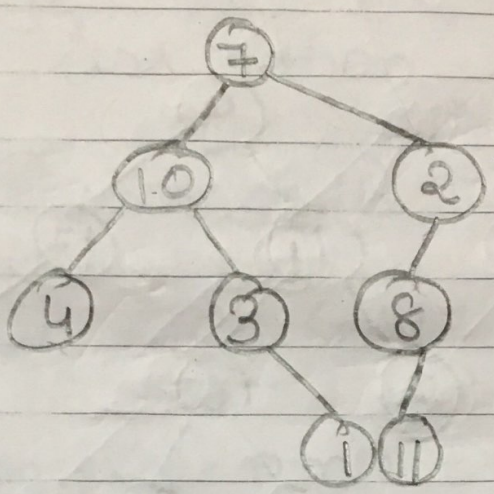
Preorder: A, B, C, D, E, F, G, H, I



Postorder: C - B - E - H - G - I - F - D - A

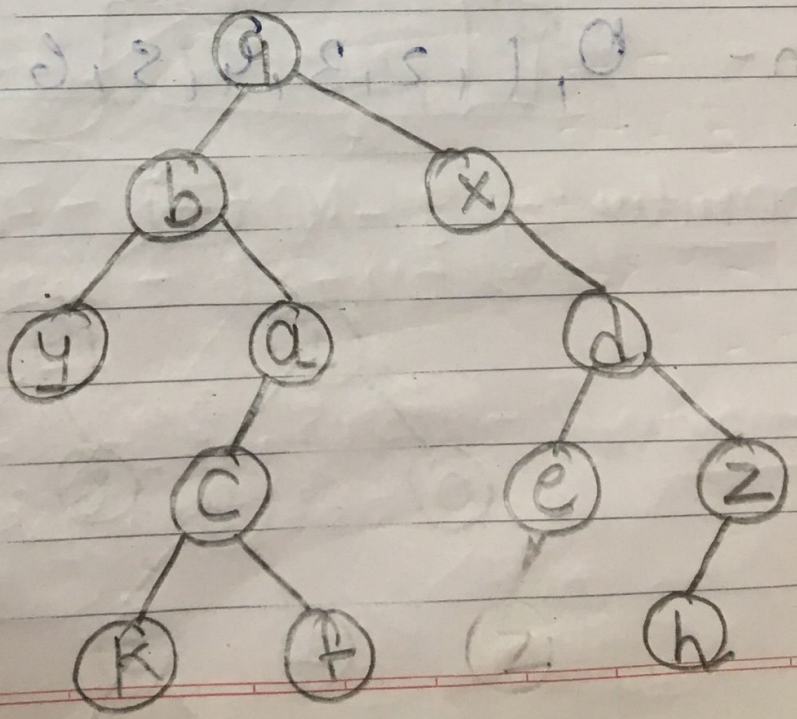
Ex: Inorder : 4, 10, 3, 1, 7, 11, 8, 2

Preorder : 7, 10, 4, 3, 1, 2, 8, 11.



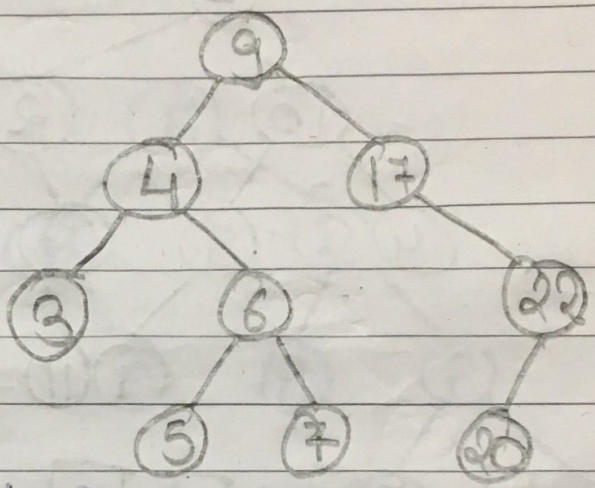
Ex: In : y, b, k, c, f, a, g, x, e, d, h, z

Pre : g, b, y, a, c, k, f, x, d, e, z, h



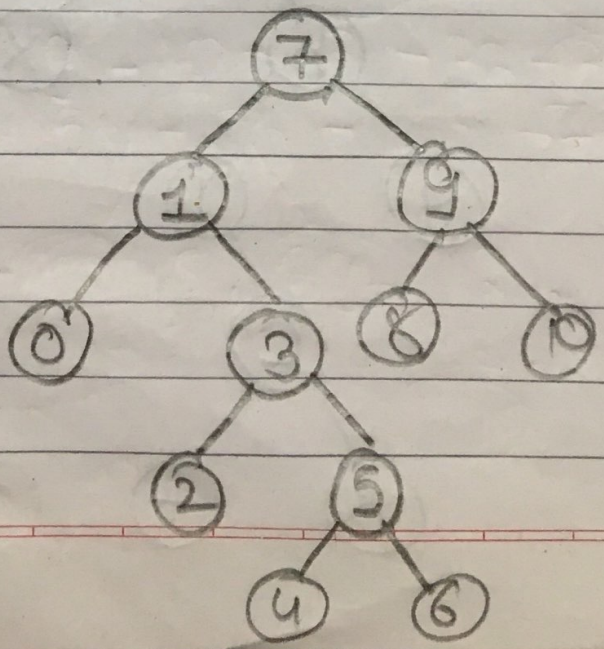
Ex: In - 3, 4, 5, 6, 7, 8, 17, 20, 22.

Pre - 9, 4, 3, 6, 5, 7, 17, 22, 20



Ex: Pre :- 7, 1, 0, 3, 2, 8, 4, 6, 9, 5, 10

In - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.



20/9/19

Date
Page

* Binary Search tree (BST)

or BST is a binary tree in which left node is less than Parent node less than right node

$$l < Parent < R$$

or Opuation On BST

=> Insertion Of node

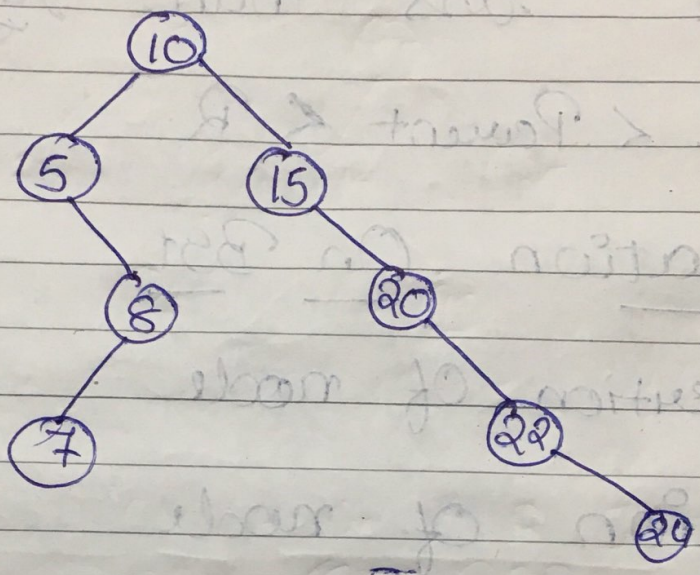
=> Deletion of node

=> Searching

=> Display of Binary tree

* Insertion

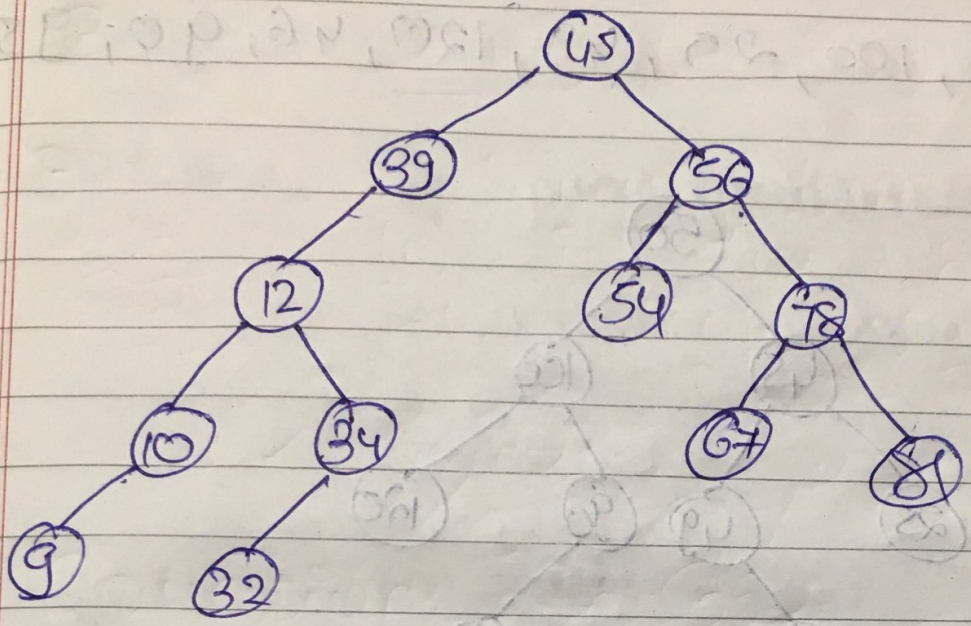
Ex: 10, 5, 15, 20, 22, 24, 8, 7



Ex: 45, 56, 39, 12, 34, 78, 54, 67, 10, 32, 9, 81

Inorder :- 9 - 10 - 12 - 32 - 34 - 39 - 45 - 54 - 56 - 67 - 78 - 81

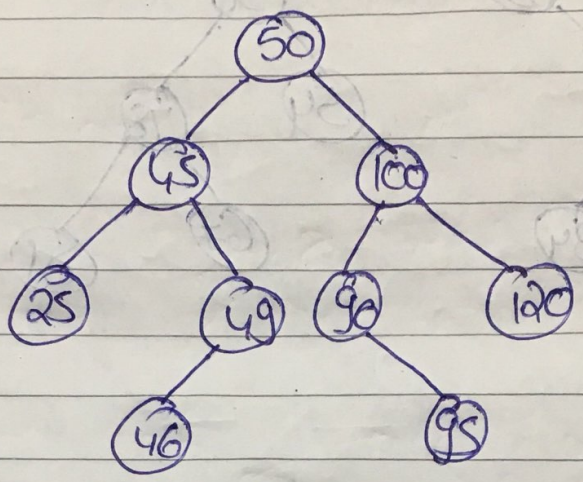
Preorder :- 45, 39, 12, 10, 9, 34, 32, 56, 54, 78, 67, 81



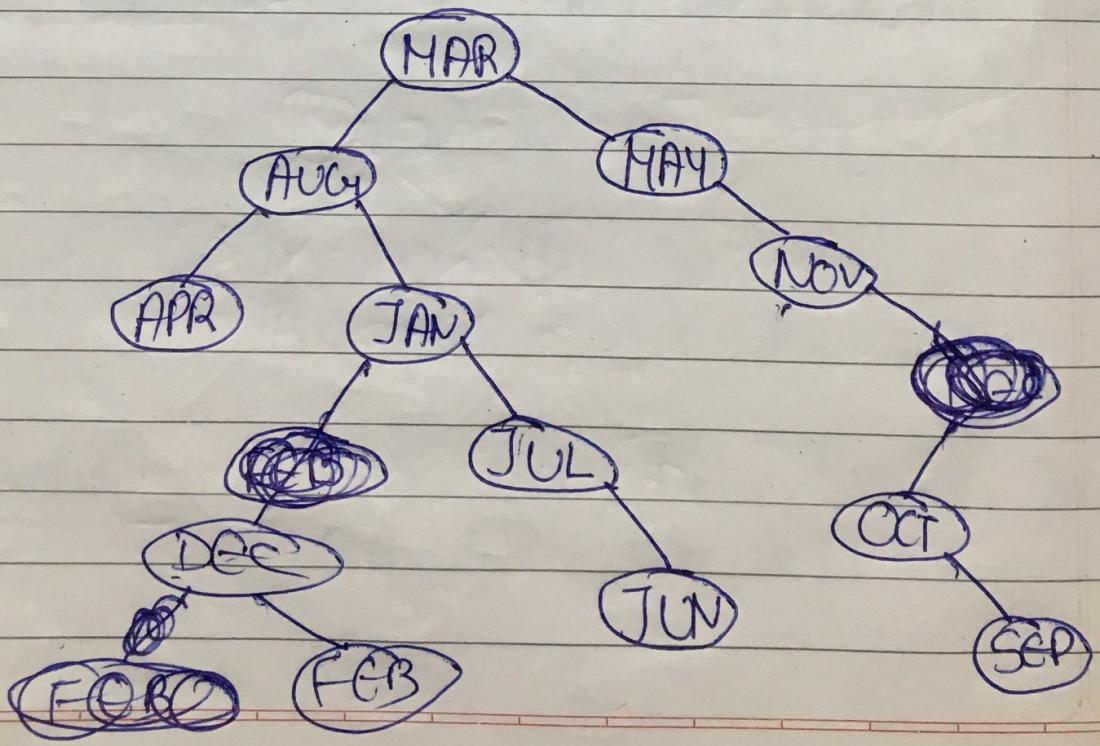
Post Order :- 9, 10, 32, 34, 12, 39, 54, 67, 81, ~~78~~, ~~56~~, 45

T

Ex: 50, 45, 100, 25, 49, 120, 46, 90, 95



Ex: MAR, MAY, NOV, AUG, APR, JAN, DEC, JUL, FEB, JUN, OCT, SEP



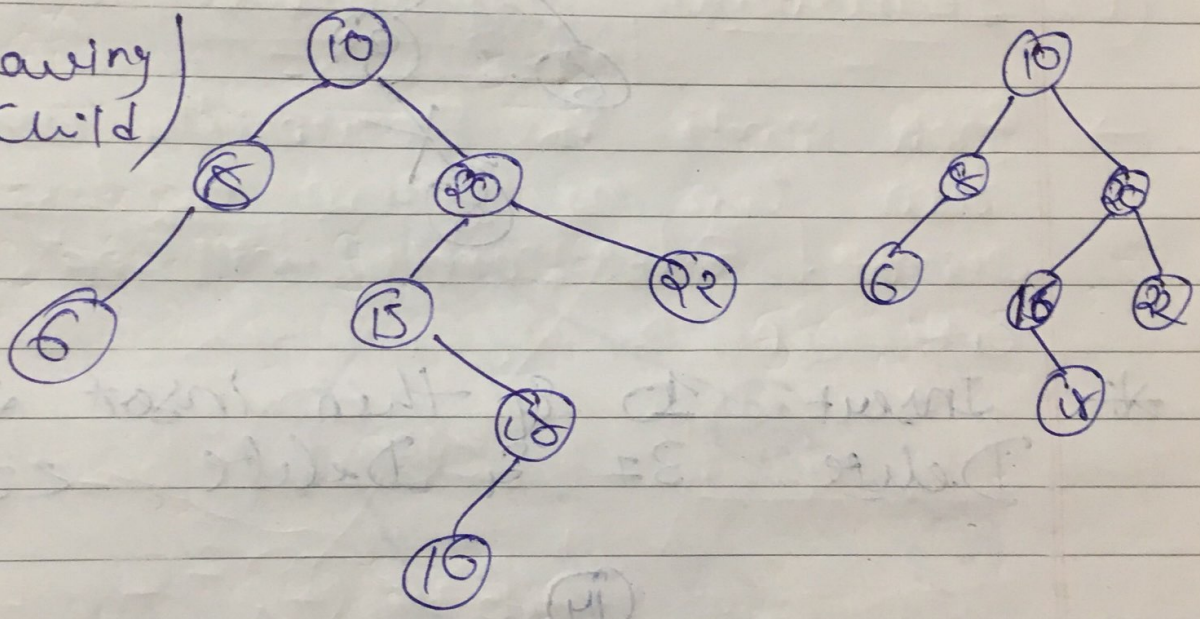
* Deletion

(1) Deletion of leaf node

(2) " " node having 1 child

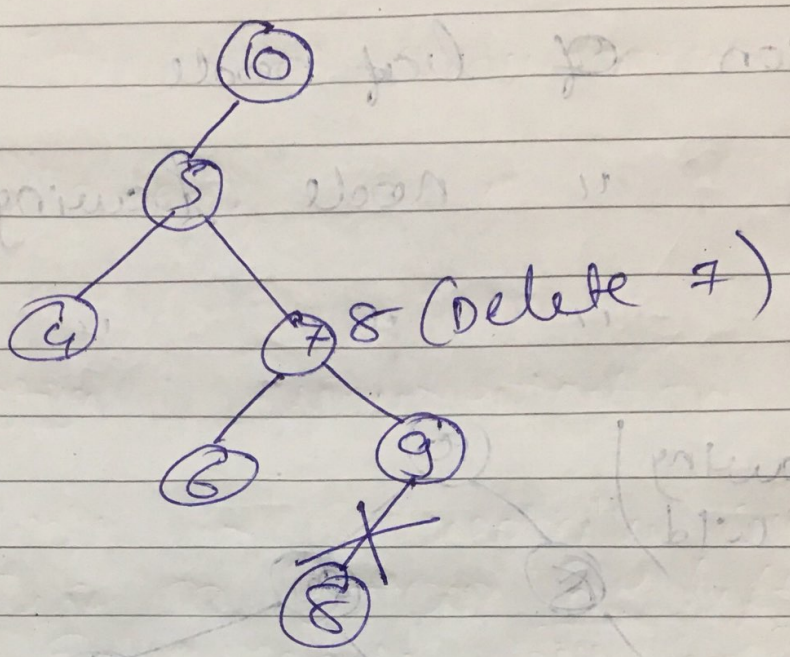
(3) " " " " 2 "

(node having one child)

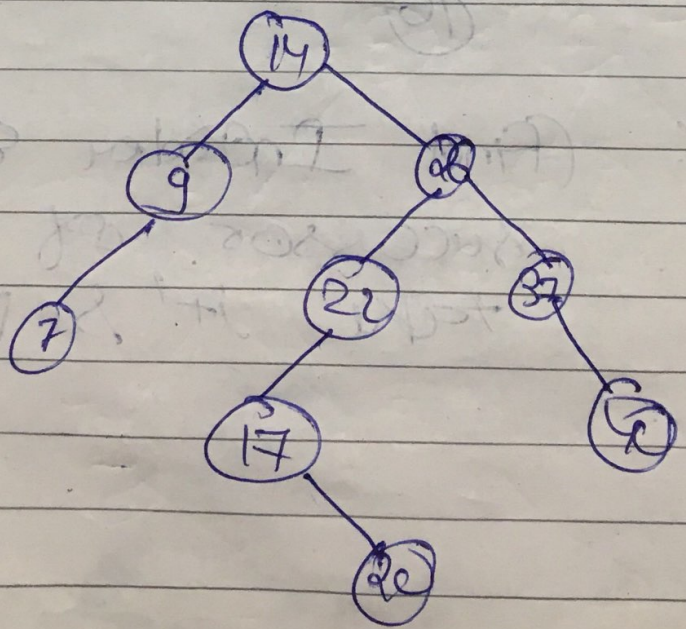


Delete is: (find Inorder of no. successor of is will take it's place)

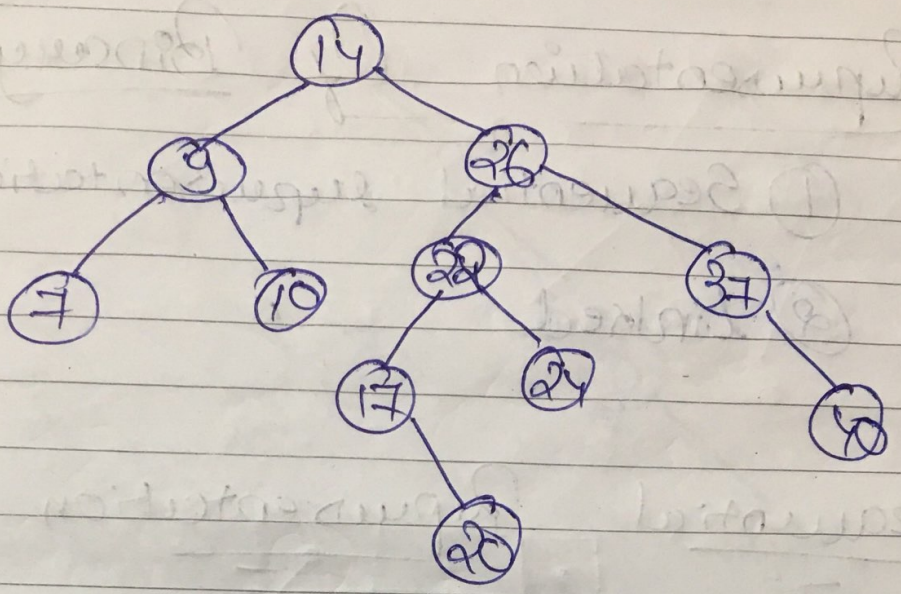
(node having 2 child)



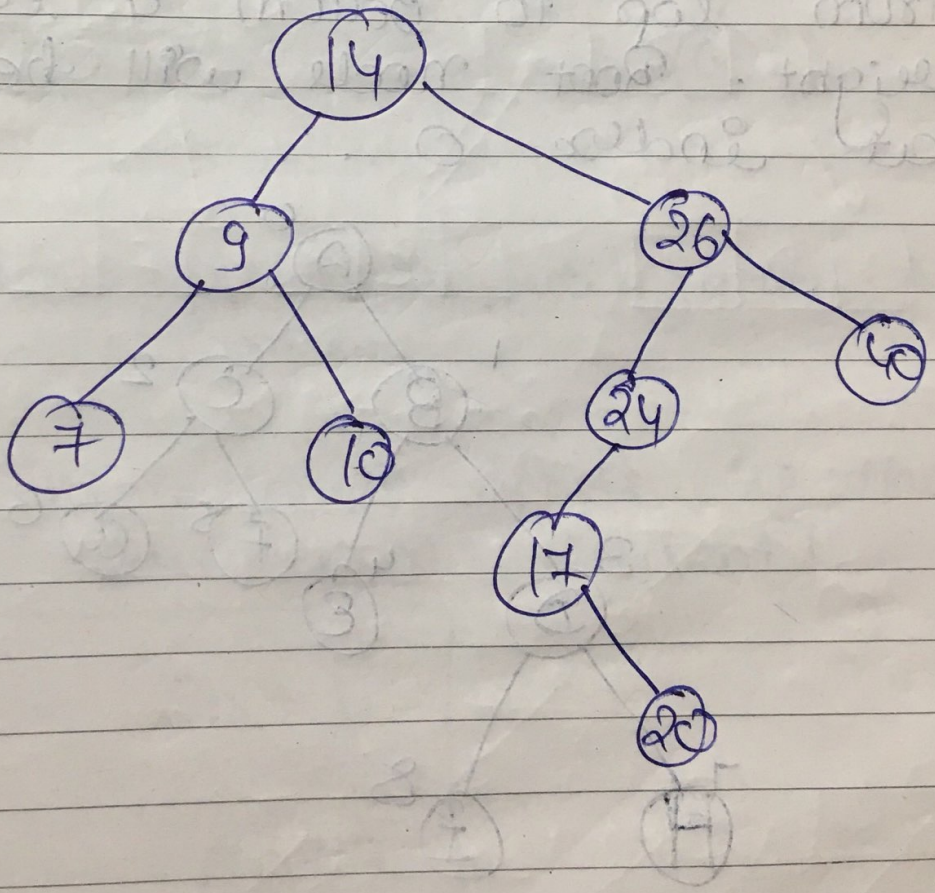
* Insert 10 & then insert 24.
Delete 37 & Delete 22



Insert



Delete



21/9/19

Date
Page

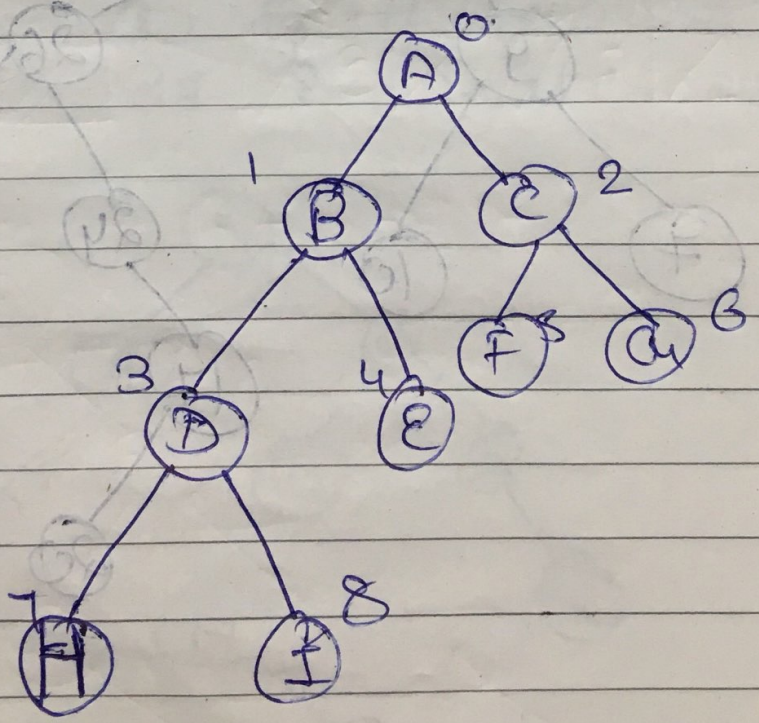
* Representation Of Binary Tree

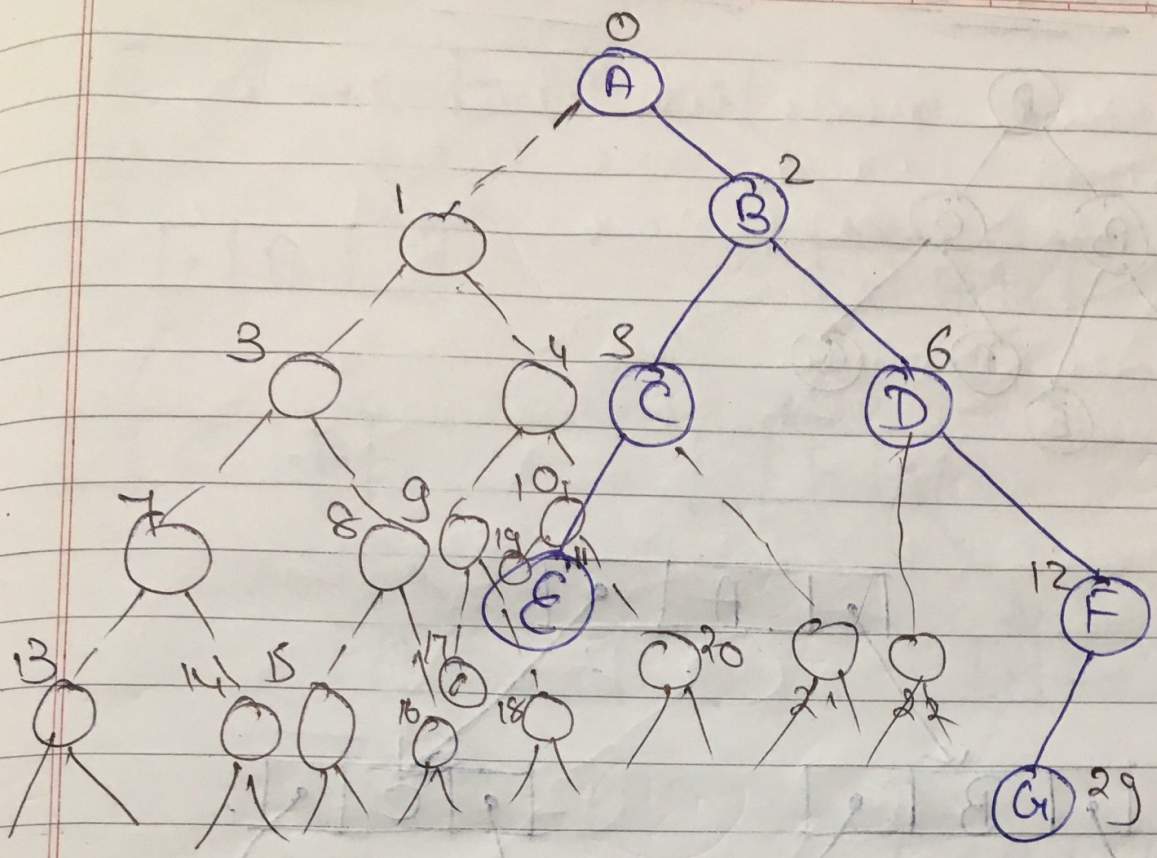
(1) Sequential representation

(2) Linked "

(1) Sequential Representation

→ Each node is sequentially arranged from top to Bottom & left to right. Root node will be always at index 0.





Disadv. memory utilization is not Proper -

[have to allocate space to those who are not present]

left - $2n+1$

right - $2n+2$