TREE

Date : 08/12/2021

Tree is basically a non-linear and hierarchial data structure.

Binary Tree :—

Every node has three fields. The first field represents the data. The second and the third field represents the address information of the left and the right child respectively.

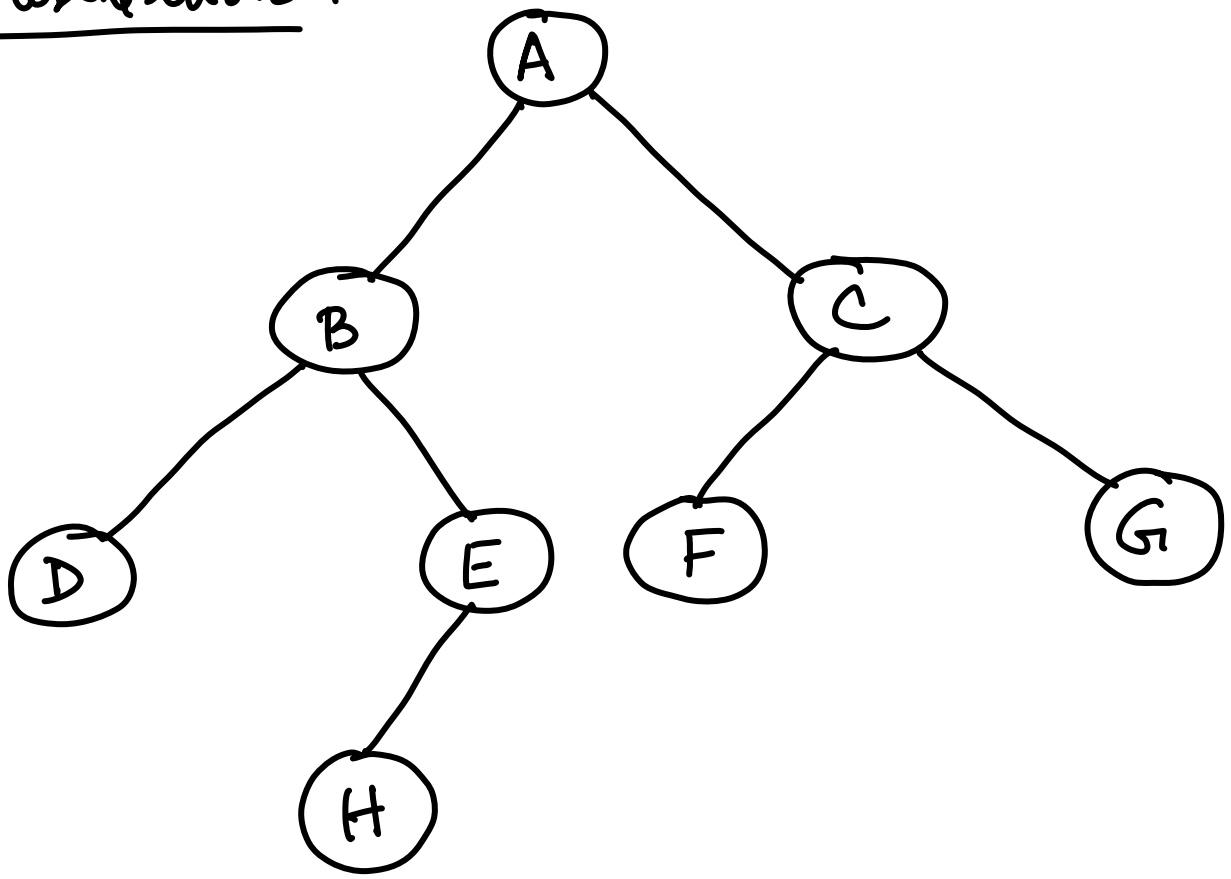
Every tree starts with a "root" node.

A Binary tree can be defined as a finite set of nodes which is either

(i) is empty

or (ii) consists of a node called "root" with two disjoint binary trees called the "left-subtree" and the "right-subtree".

Representation :



Here, A is the "root" B is the left "child / successor", C is the right "child / successor" of A. This makes A as the "father" of both B and C. Similarly, D and E are the left and "right child" of B respectively. In other words, B is the "father" of both D and E.

Any node shall be termed as the "leaf node / terminal node" if it

has no children.

Here, D, H, F and G nodes are the leaf nodes / terminal nodes.

If two nodes have the same father then they are termed as "siblings or brother nodes". So, B and C are siblings. Similarly, D and E are siblings Also, we find F and G are siblings.

The nodes other than the terminal/leaf nodes, are termed as "internal" nodes of a tree. Here, A, B, C, E are internal nodes.

The "level" of every node in a binary tree can be defined as following -

(i) the root resides in level 0

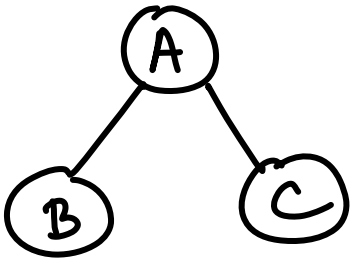
(ii) the level number of other nodes is 1 more than the level number of its father node

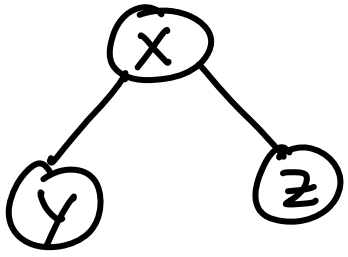
So, here we find

node	level
A	0
B , C	1
D , E , F , G	2
H	3

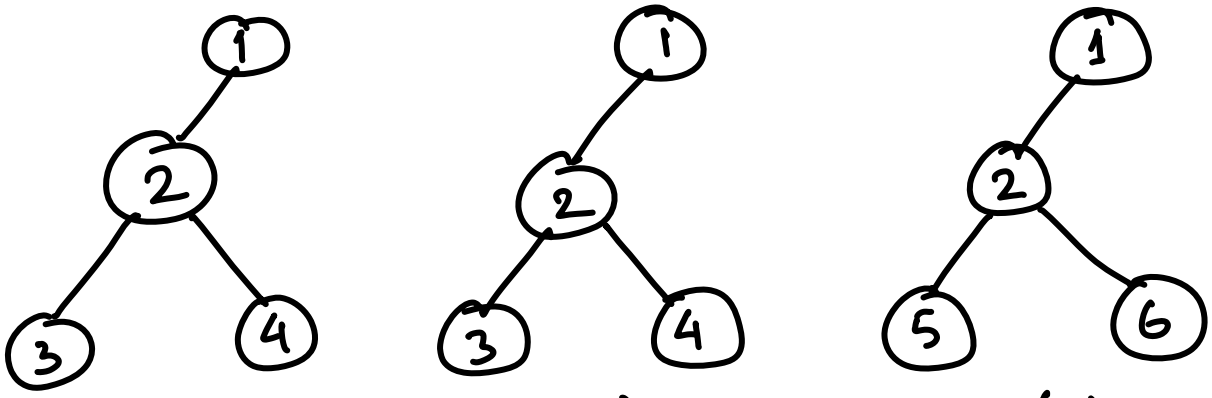
The "height / depth" of a tree is 1 more than its largest level number. So, the height of our example binary tree is $3+1 = 4$.

Two trees can be called "similar" if they have similar data structure and also said to be "copies" if they have same data at same node positions.





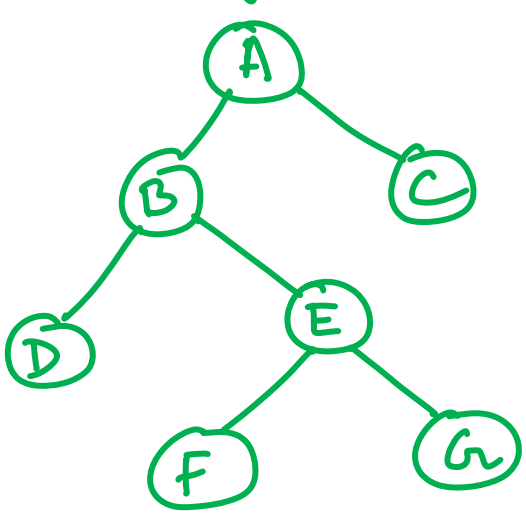
These above two trees are called "similar" trees.



tree-(a) and tree-(b) are "copies" of one another. Where as, tree-(c) and tree-(a) or in other words, tree-(c) and tree-(b) are termed as similar trees.

Some Types of Binary Trees

(1) Strictly Binary Tree -

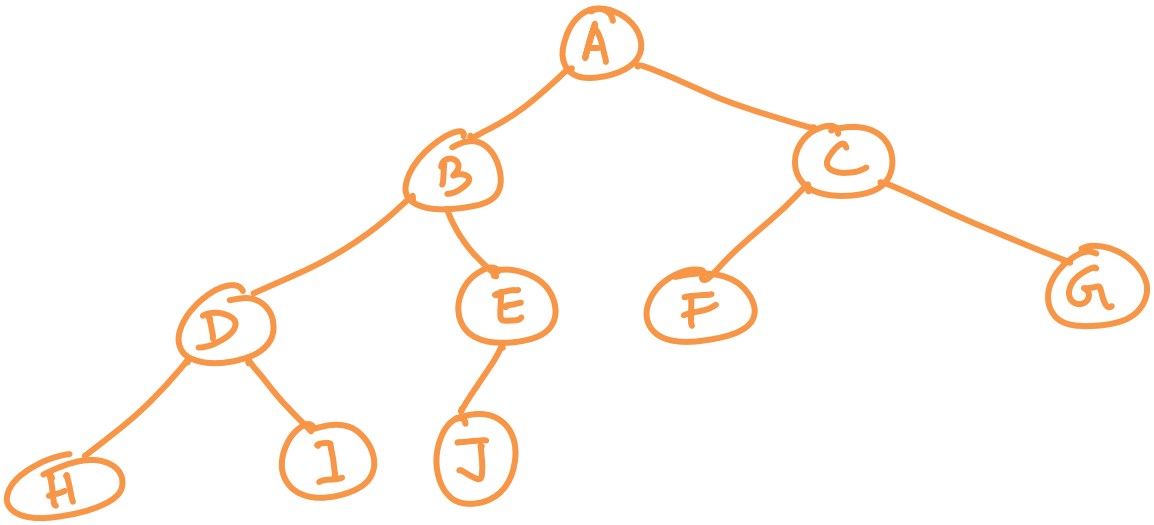


Every internal node has its left as well as right child.

(2) Complete Binary Tree

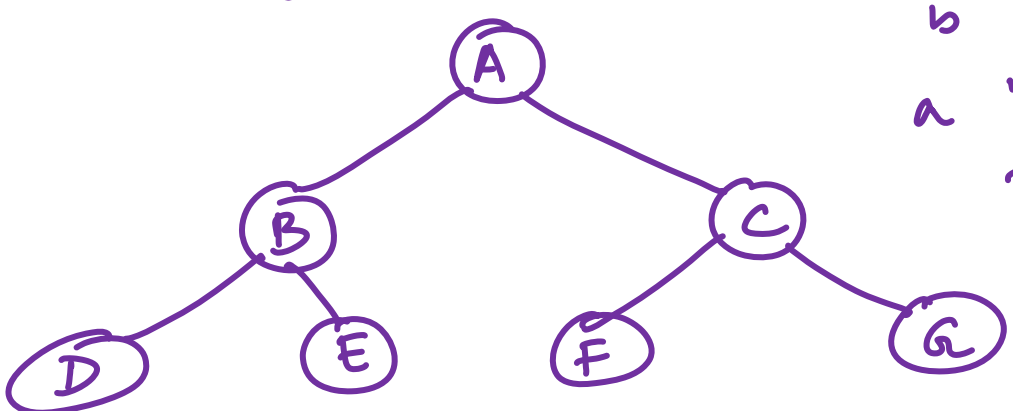
defined as (i) all leaf-nodes must be situated in adjacent levels.

(ii) the child positions of any non-leaf node / internal node will be filled up in a left-to-right sequence.



(3) Fully-Complete Binary Tree

is also called a "Full-binary" tree.

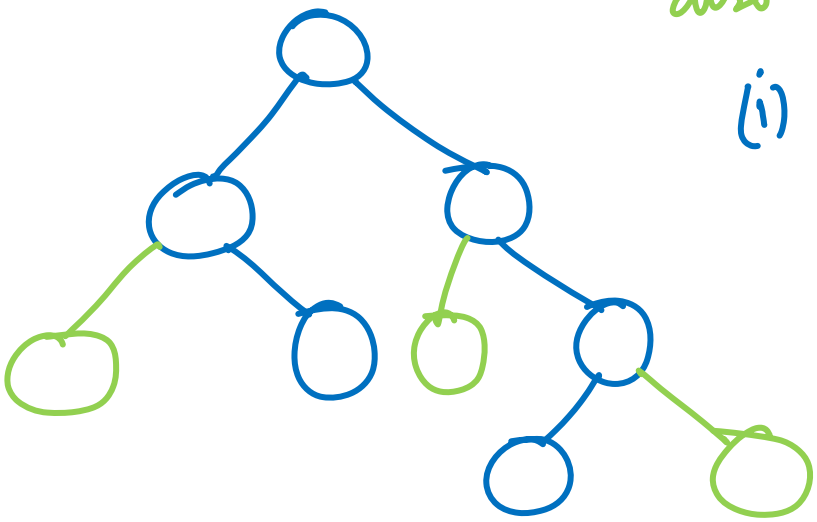


(i) all internal nodes have two children

(ii) all leaf nodes are in the same level

(4) Extended Binary tree

also called a "2-tree"



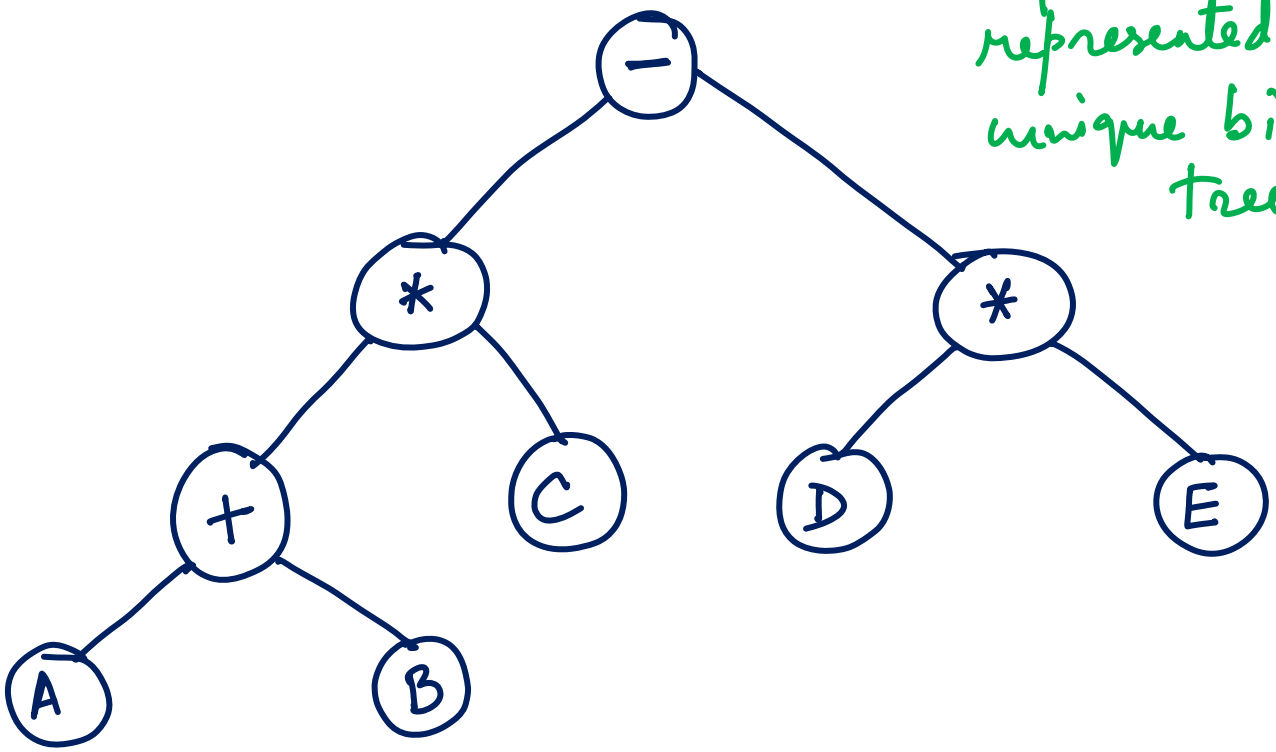
(i) every node of such a tree has zero or two children.

Representation of a Mathematical expression using a Binary Tree

Dated on 09/12/2021

$(A+B)*C-(D*E)$

Every algebric expression is represented by a unique binary tree.

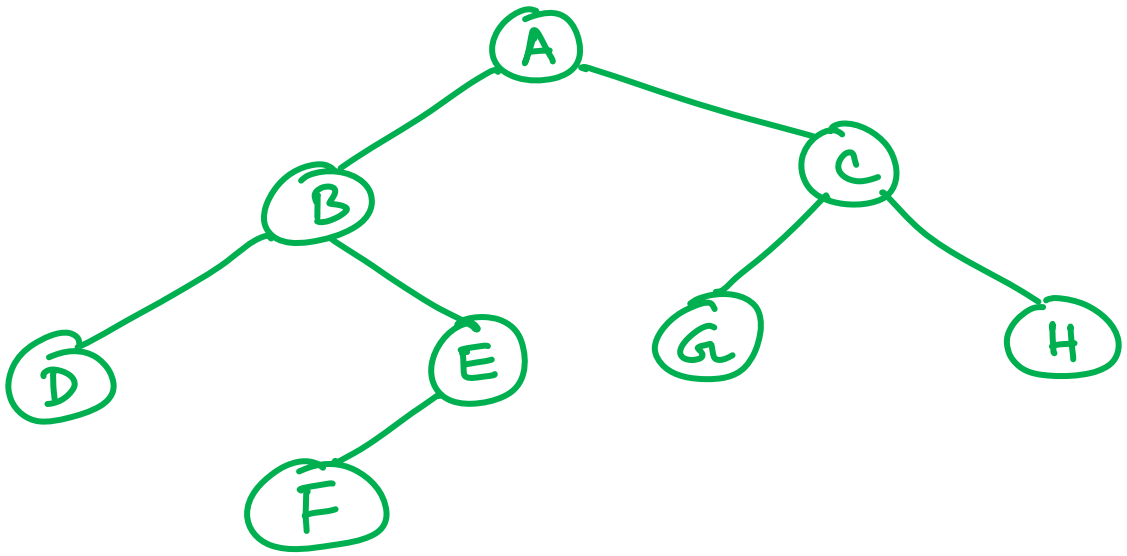


Representation of a Binary Tree

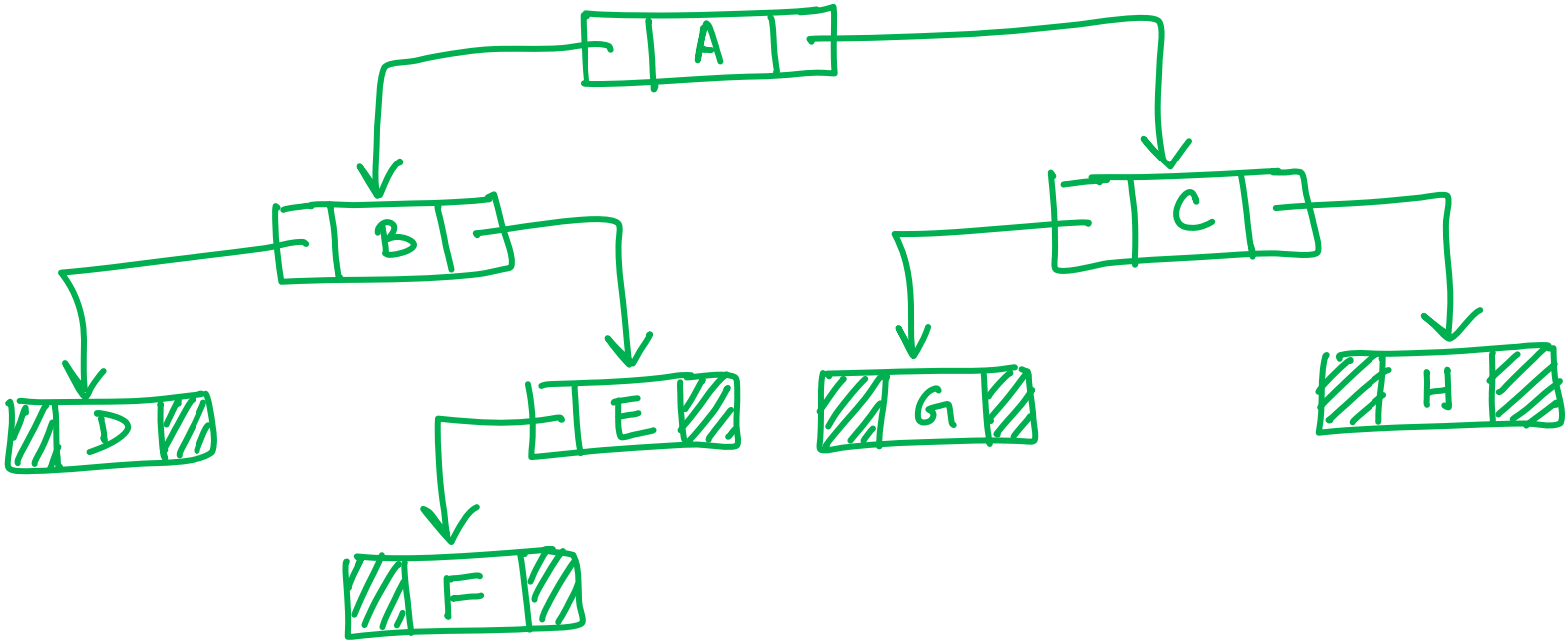
(1) Linked list representation

(2) Array representation

Linked list representation of binary tree :-




```
struct node
    {
        char data;
        struct node * lchild;
        struct node * rchild;
    };
```



Binary Tree Traversal :-

(1) Visit the root first, then visit the left-subtree and then the right-subtree (NLR Traversal)

Preorder traversal

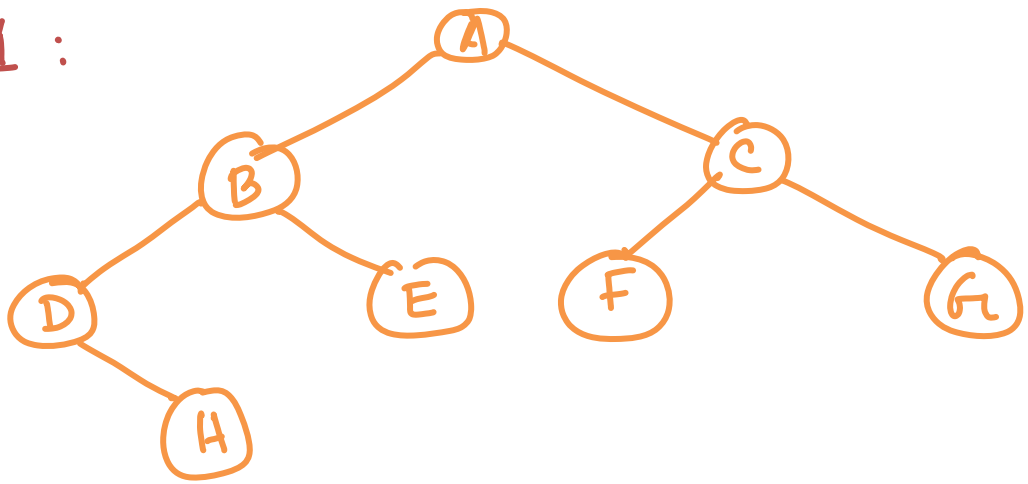
(2) Visit the left-subtree first, then visit the root and then the right-subtree. (LNR Traversal)

Inorder traversal

(3) Visit the left-subtree first, then visit the right-subtree and then visit the root. (LRN Traversal)

Postorder traversal

Example-1 :

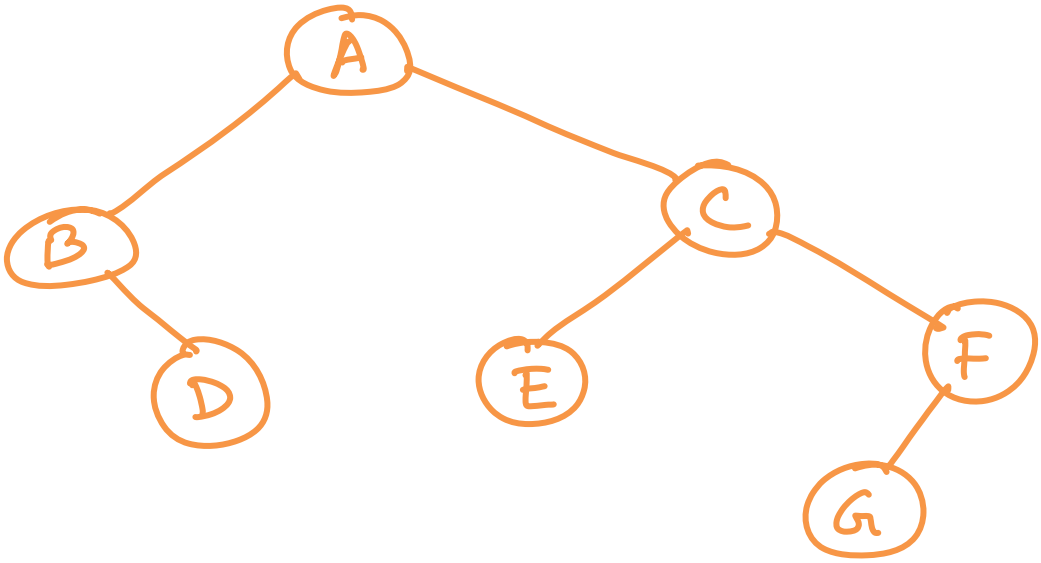


Preorder traversal (NLR): A, B, D, H, E, C, F, G

Inorder traversal (LNR): D, H, B, E, A, F, C, G

Postorder traversal (LRN): H, D, E, B, F, G, C, A

Example-2 :



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

Inorder : B, D, A, E, C, G, F

Postorder : D, B, E, G, F, C, A