DATA STRUCTURE AND ALGORITHMS

By

A.ROGHYA PARVEEN

ASSISTANT PROFESSOR IN DEPARTMENT OF CS AND IT

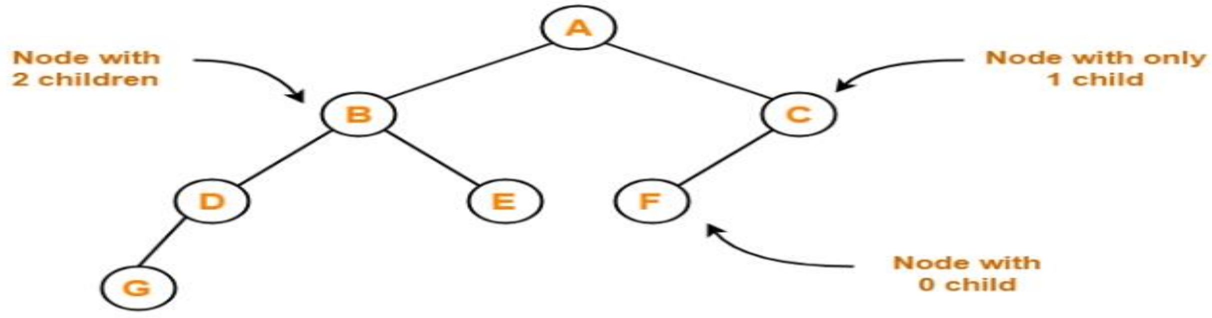
JAMAL MOHAMED COLLEGE(AUTONOMOUS), TRICHY-20.

UNIT II
TREES

BINARY TREE

Binary tree is a special tree data structure in which each node can have at most 2 children.

Thus, in a binary tree,Each node has either 0 child or 1 child or 2 children.

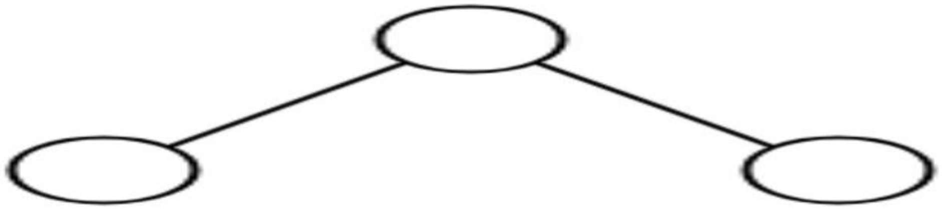


Binary Tree Example

Types of Binary Tree

Unlabeled Binary Tree-

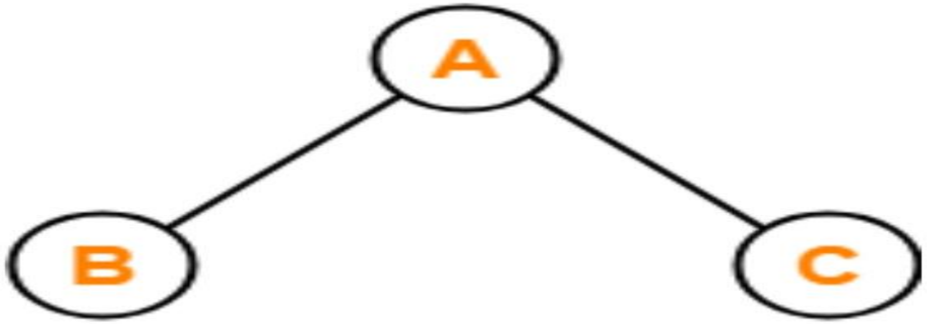
A binary tree is unlabeled if its nodes are not assigned any label.



Unlabeled Binary Tree

Labeled Binary Tree–

- A binary tree is labeled if all its nodes are assigned a label.



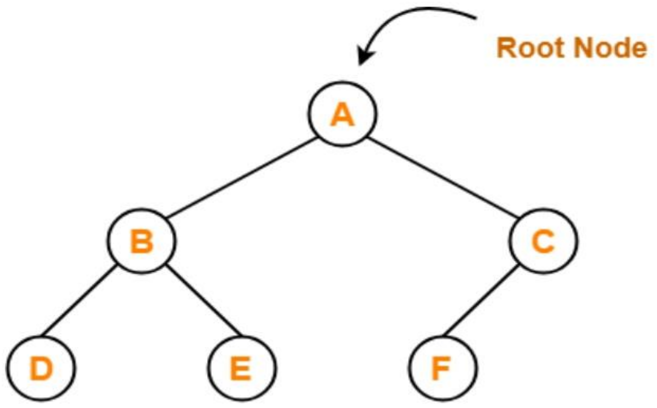
Labeled Binary Tree

Types of Binary Tree

1. Rooted Binary Tree
2. Full / Strictly Binary Tree
3. Complete / Perfect Binary Tree
4. Almost Complete Binary Tree
5. Skewed Binary Tree

1. Rooted Binary Tree-

- A **rooted binary tree** is a binary tree that satisfies the following 2 properties-
- It has a root node.
- Each node has at most 2 children.

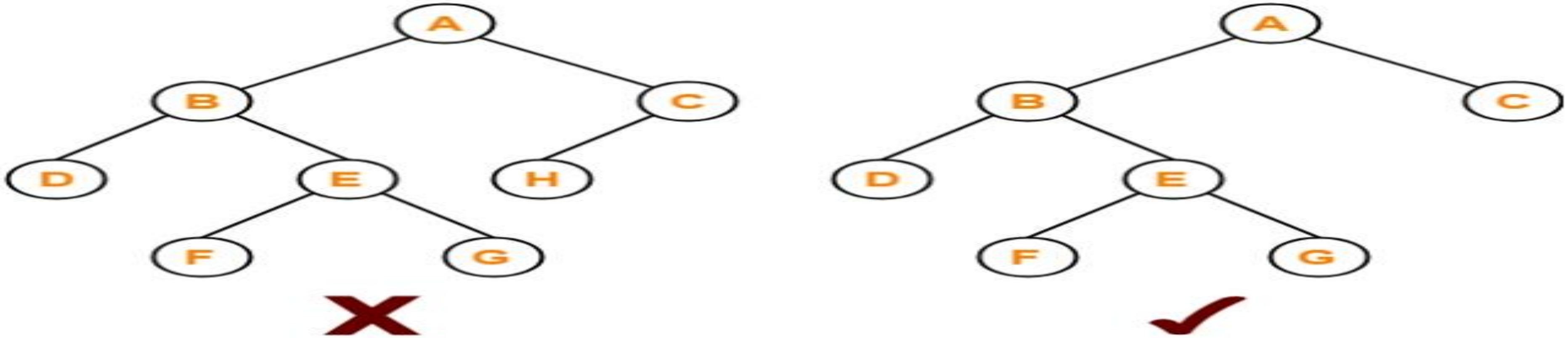


Rooted Binary Tree

2 . Full / Strictly Binary Tree–

- A binary tree in which every node has either 0 or 2 children is called as a **Full binary tree**.
- Full binary tree is also called as **Strictly binary tree**.

Example-

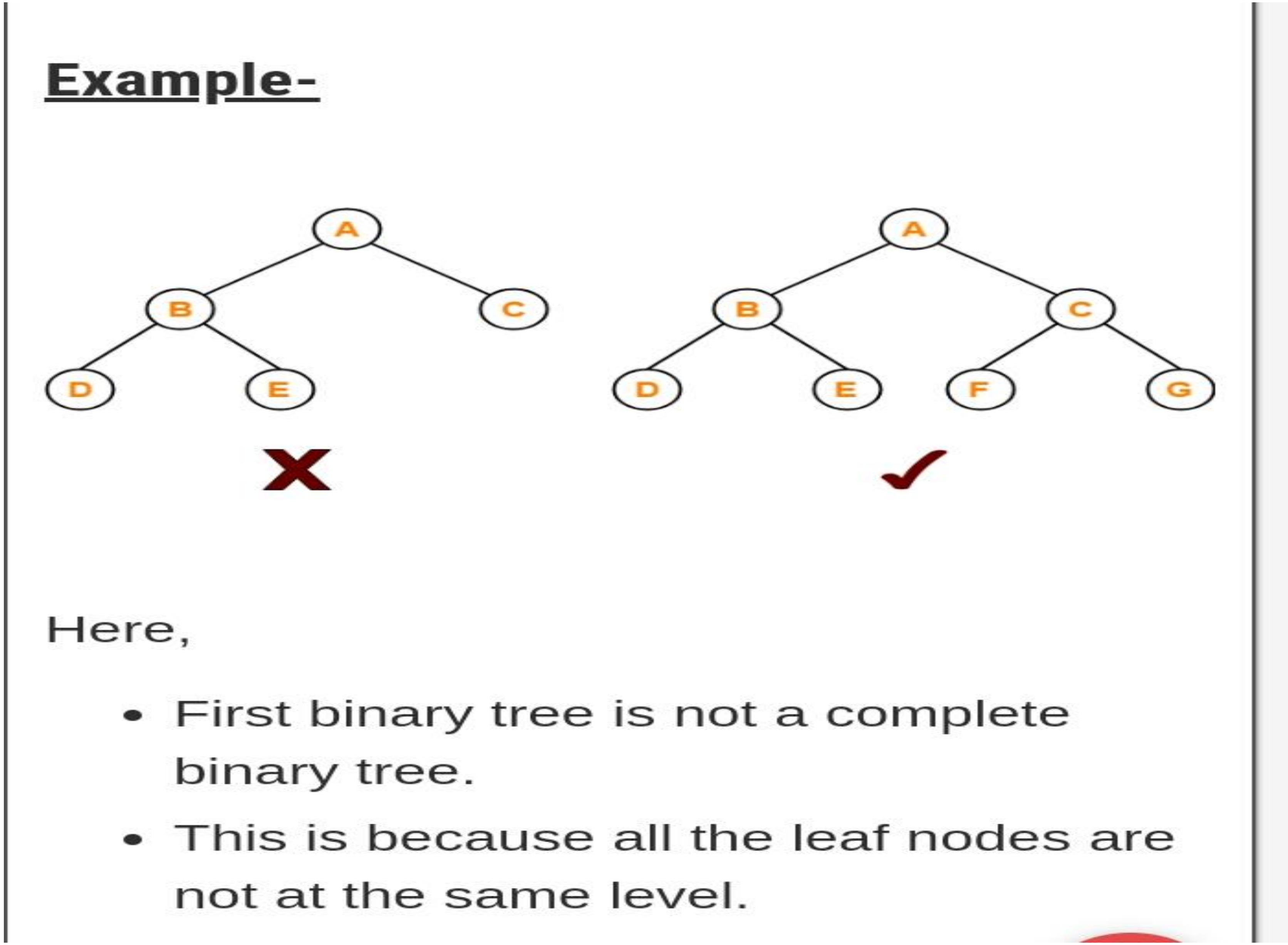


Here,

- First binary tree is not a full binary tree.
- This is because node C has only 1 child.

3. Complete / Perfect Binary Tree

- A **complete binary tree** is a binary tree that satisfies the following 2 properties-
- Every internal node has exactly 2 children.
- All the leaf nodes are at the same level.
- Complete binary tree is also called as **Perfect binary tree**.



Example-

Here,

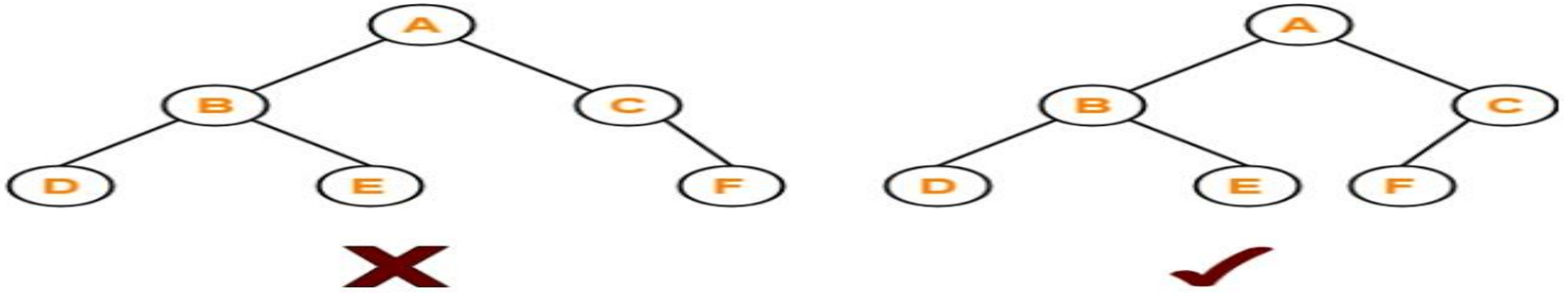
- First binary tree is not a complete binary tree.
- This is because all the leaf nodes are not at the same level.

4. Almost Complete Binary Tree–

An **almost complete binary tree** is a binary tree that satisfies the following 2 properties-

- All the levels are completely filled except possibly the last level.
- The last level must be strictly filled from left to right.

Example-



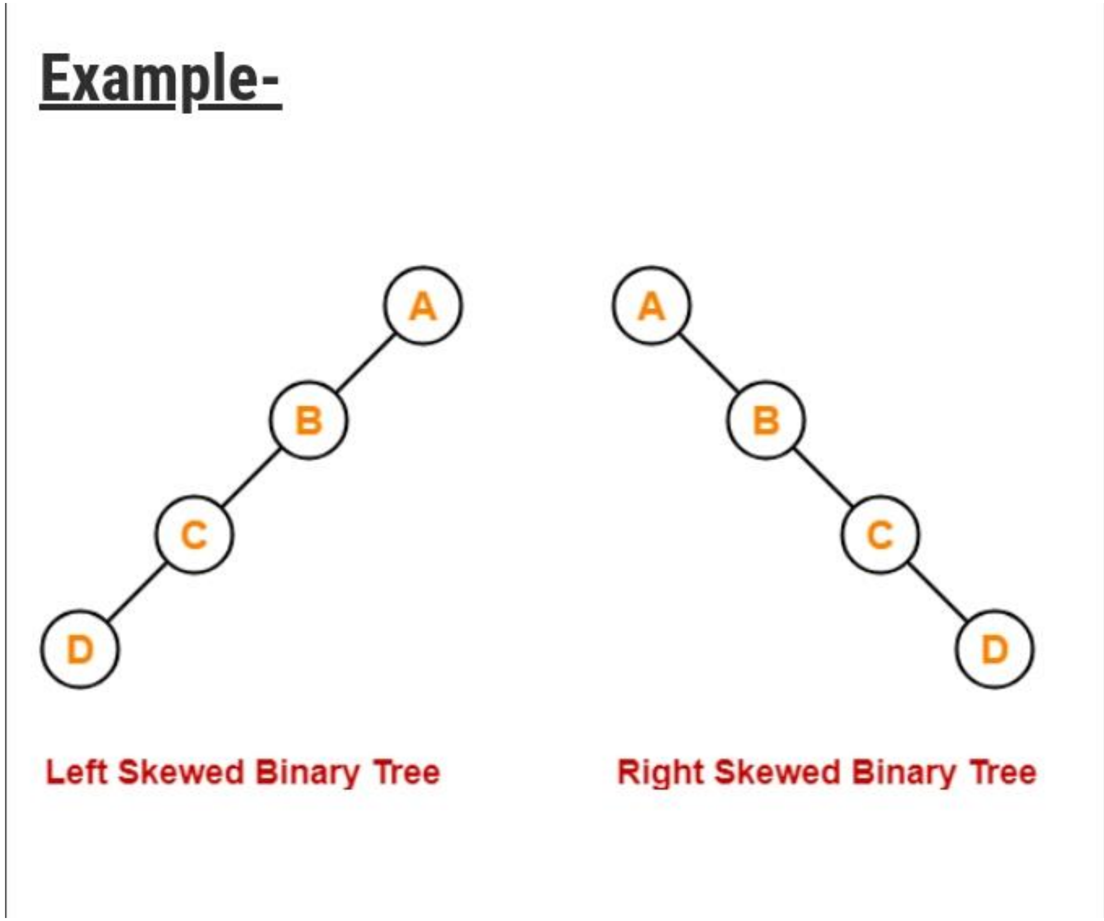
Here,
First binary tree is not an almost complete binary tree.
This is because the last level is not filled from left to right.

5. Skewed Binary Tree

- A **skewed binary tree** is a binary tree that satisfies the following 2 properties-
- All the nodes except one node has one and only one child.
- The remaining node has no child.

OR

- A **skewed binary tree** is a binary tree of n nodes such that its depth is (n-1).



Example-

Left Skewed Binary Tree

Right Skewed Binary Tree

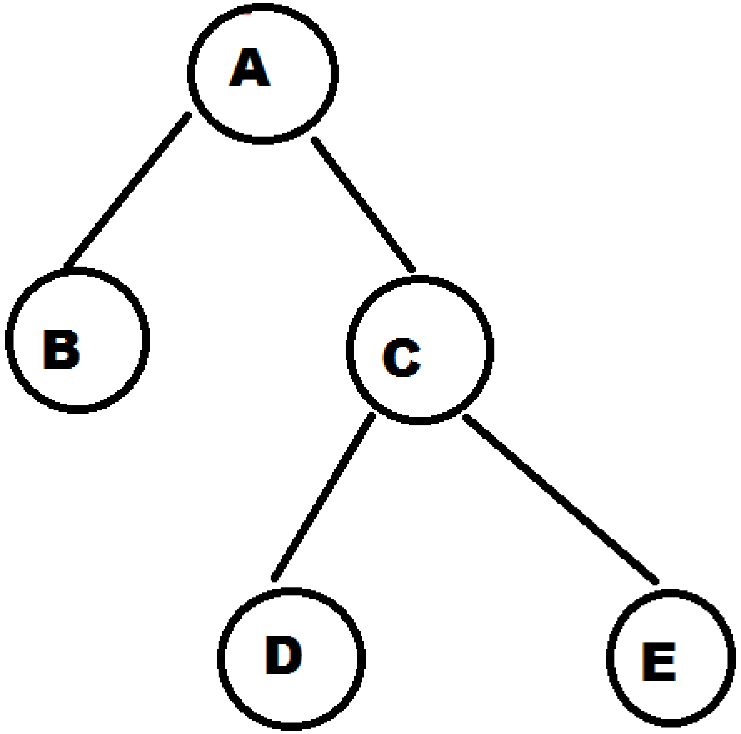
Binary Tree representation

There are two types of representation of a binary tree:

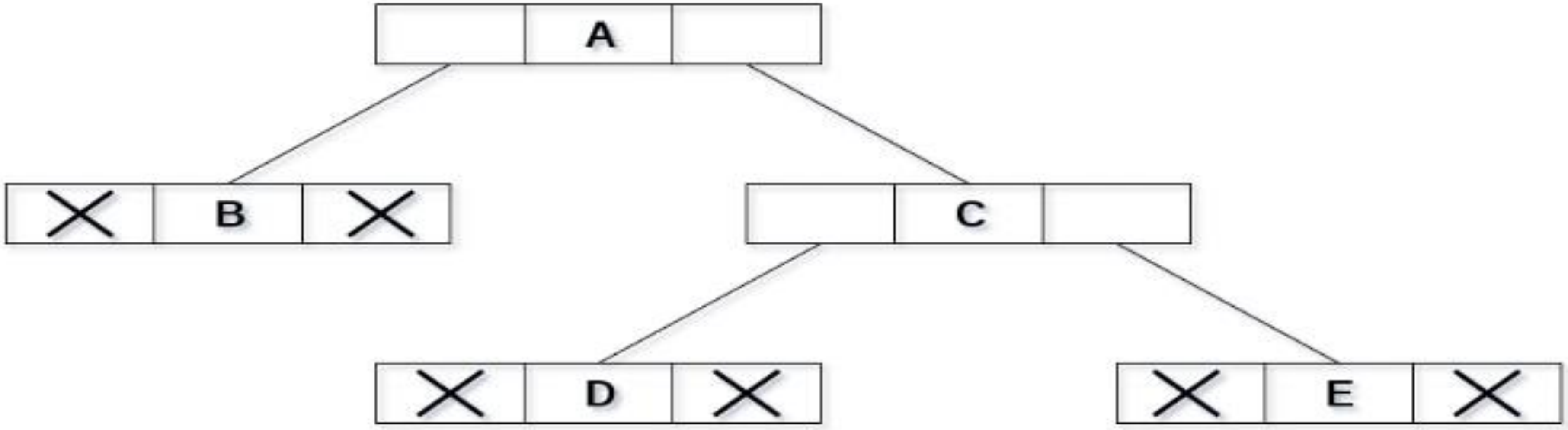
1. Linked Representation

- In this representation, the binary tree is stored in the memory, in the form of a linked list where nodes are linked together by inheriting parent child relationship like a tree.

For this binary tree



Linked Representation is

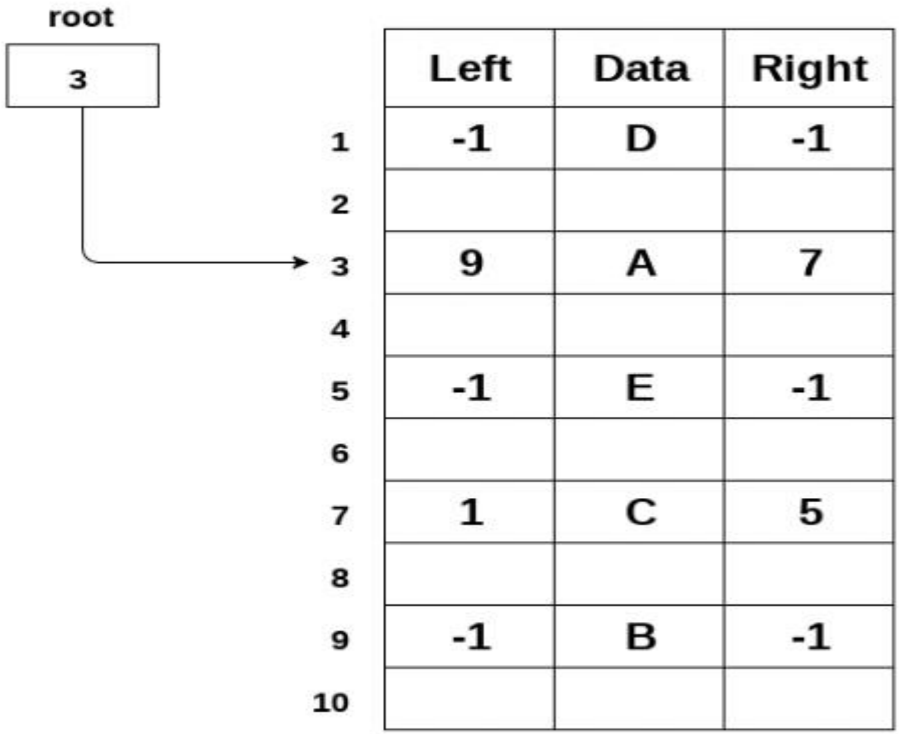


In the above figure, a tree is seen as the collection of nodes where each node contains three parts :

- left pointer,
- data element and
- right pointer.
- Left pointer stores the address of the left child while the right pointer stores the address of the right child.
- The leaf node contains **null** in its left and right pointers.

The following image shows about how the memory will be allocated for the binary tree by using linked representation.

- There is a special pointer maintained in the memory which points to the root node of the tree.
- Every node in the tree contains the address of its left and right child.
- Leaf node contains null in its left and right pointers.



root

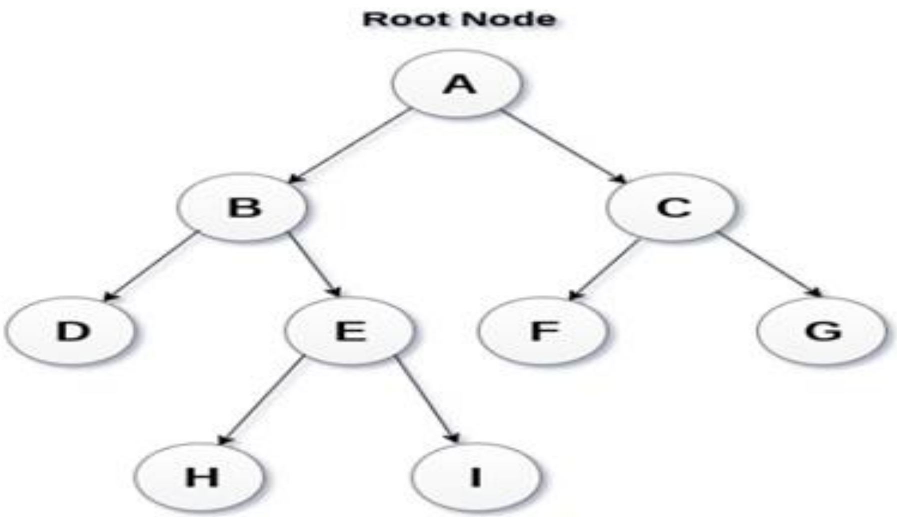
3

	Left	Data	Right
1	-1	D	-1
2			
3	9	A	7
4			
5	-1	E	-1
6			
7	1	C	5
8			
9	-1	B	-1
10			

Memory Allocation of Binary Tree using linked Representation

2.Sequential Representation

- This is the simplest memory allocation technique to store the tree elements.
- But it is an inefficient technique since it requires a lot of space to store the tree elements.
- A binary tree is shown in the following figure along with its memory allocation.



A	B	C	D	E	F	G			H	I
1	2	3	4	5	6	7	8	9	10	11

Sequential Representation of Binary Tree

2.Sequential Representation

- In this representation, an array is used to store the tree elements. Size of the array will be equal to the number of nodes present in the tree.
- The root node of the tree will be present at the 1st index of the array.
- If a node is stored at ith index then its left and right children will be stored at 2i and 2i+1 location.
- If the 1st index of the array i.e. tree[1] is 0, it means that the tree is empty.

DATA STRUCTURES AND ALGORTIHMS

BY

A.ROGHYA PARVEEN

ASSISTANT PROFESSOR IN DEPARTMENT OF COMPUTER SCIENCE AND IT

JAMAL MOHAMED COLLEGE(AUTONOMOUS),TRICHY.

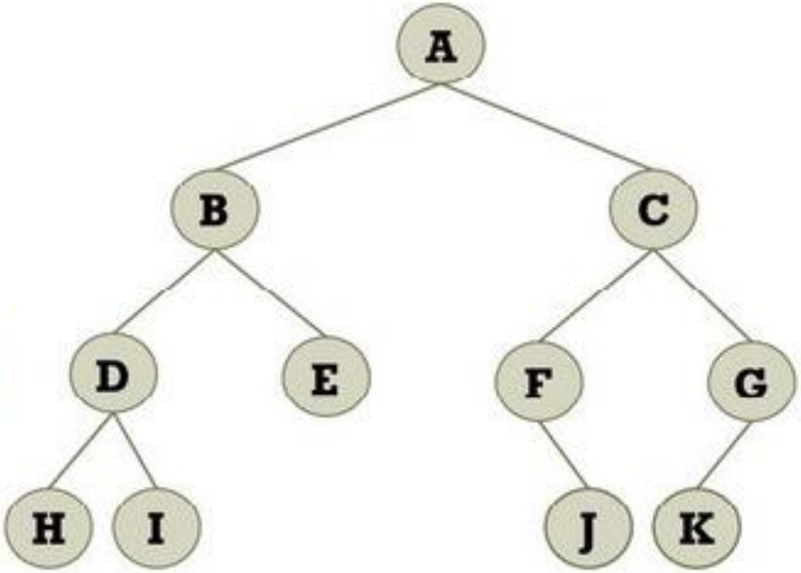
BINARY TREE TRAVERSAL

Binary Tree Traversal

Tree Traversal refers to the process of visiting each node in a tree data structure exactly once.

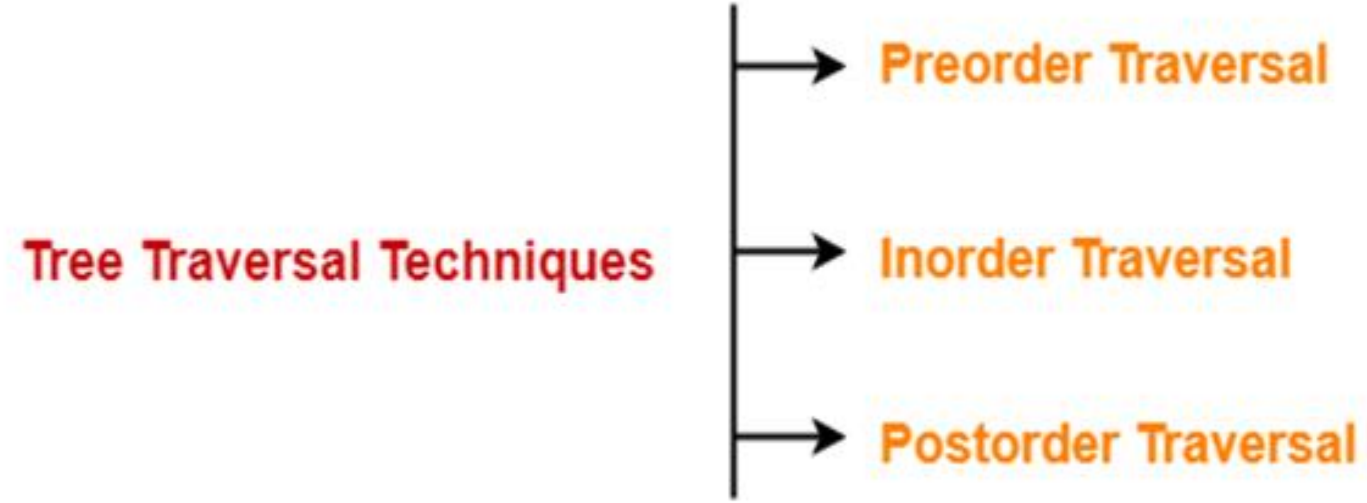
Traversal of Binary Tree

- Traversal
  - The process of visiting each node in a tree
- Why the traversal necessary?
  - Checking whether insertions/deletions work well.
  - Searching a specific node.
- How to visit all nodes once?



Binary Tree Traversal

Various tree traversal methods are-



Tree Traversal Techniques

- Preorder Traversal
- Inorder Traversal
- Postorder Traversal

1. Preorder Traversal

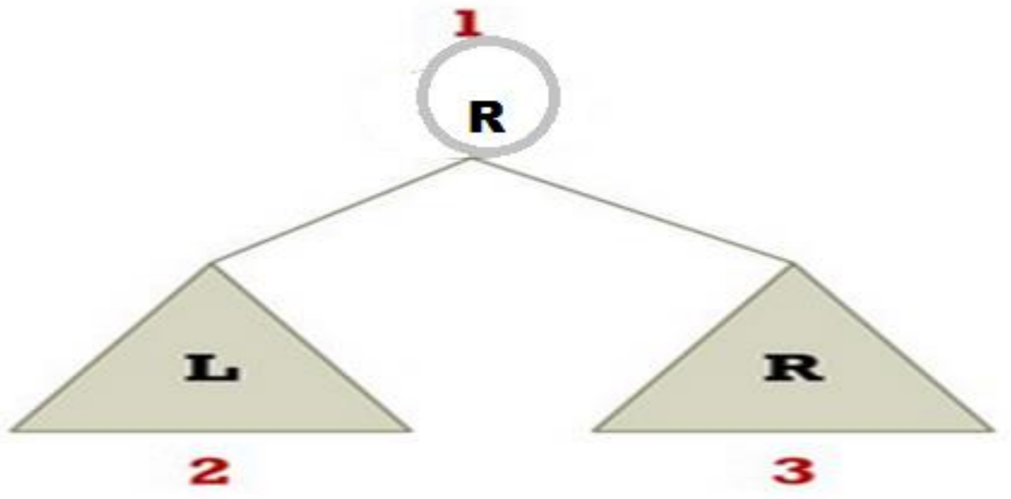
Algorithm-

1. Visit the root node.
2. Visit all the nodes in the left subtree.
3. Visit all the nodes in the right subtree

Root → Left → Right

Always remember for preorder

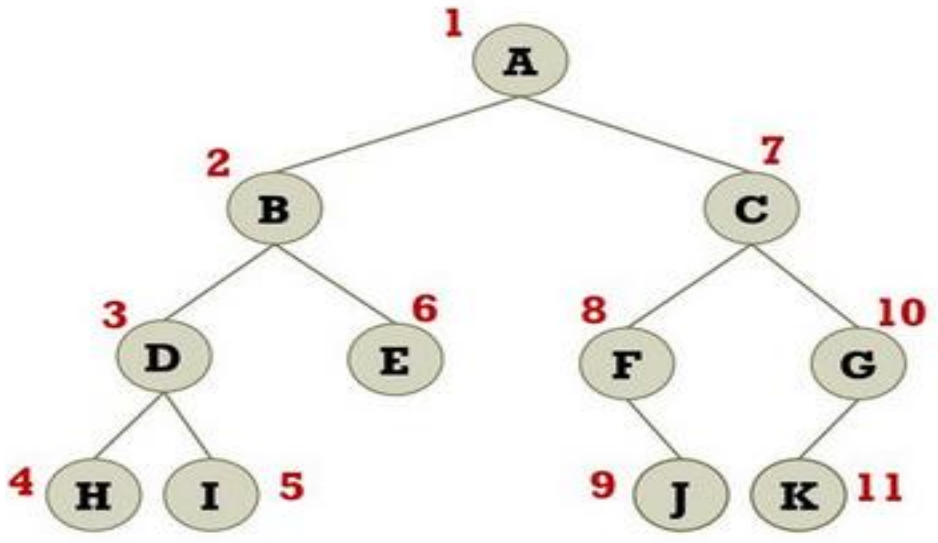
Root → Left → Right **or** **DATA--> LEFT --> RIGHT**

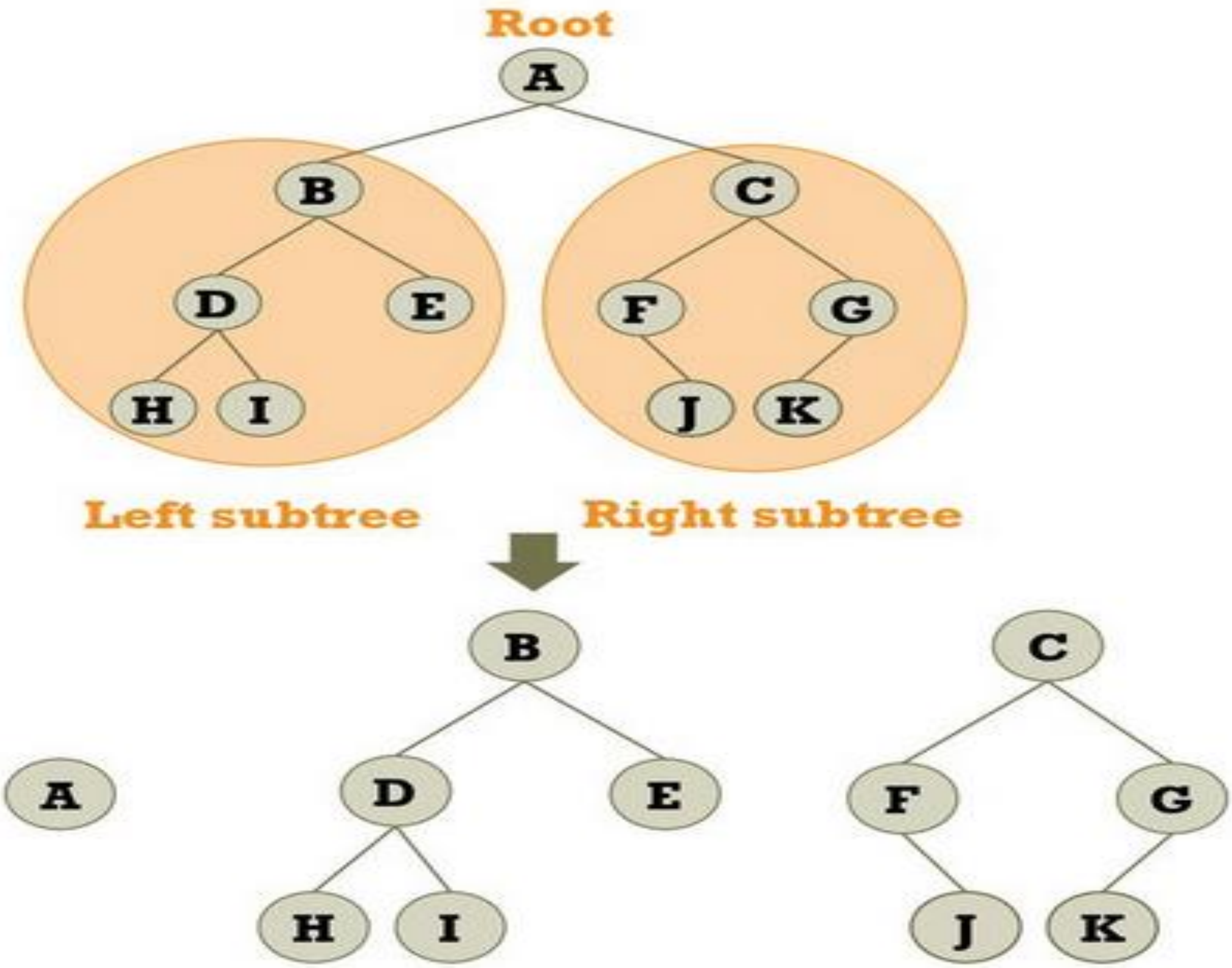


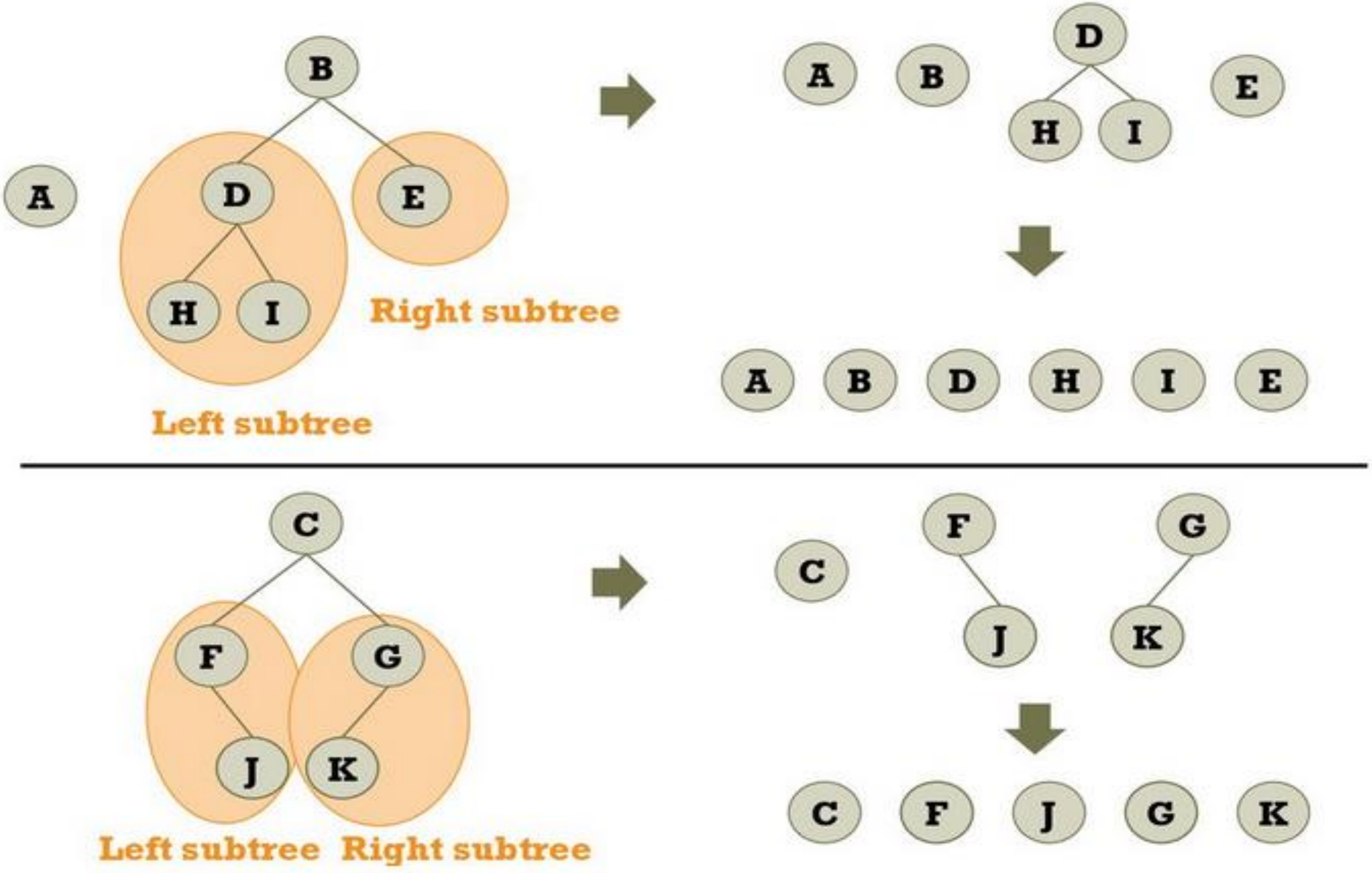
1. Preorder Traversal

Example

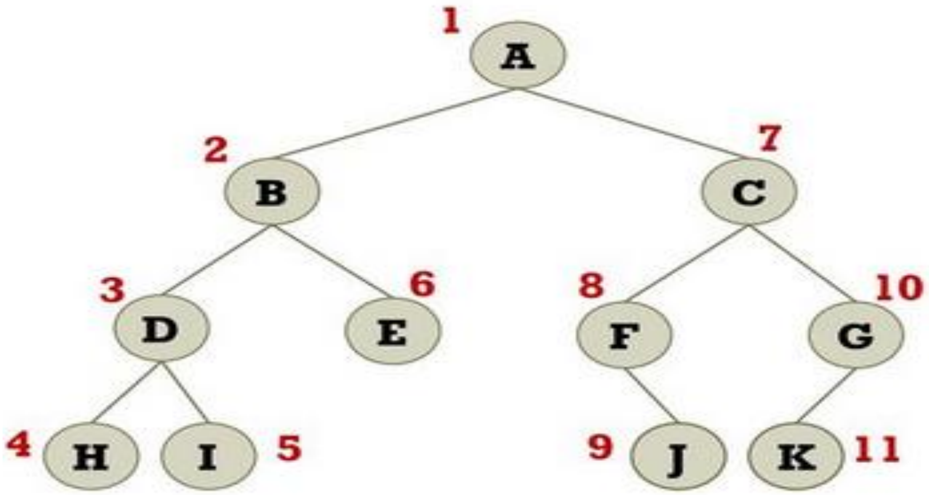
Consider the following example




Root
A
B
C
D
E
F
G
H
I
J
K
Left subtree
Right subtree
B
C
A
D
E
F
G
H
I
J
K

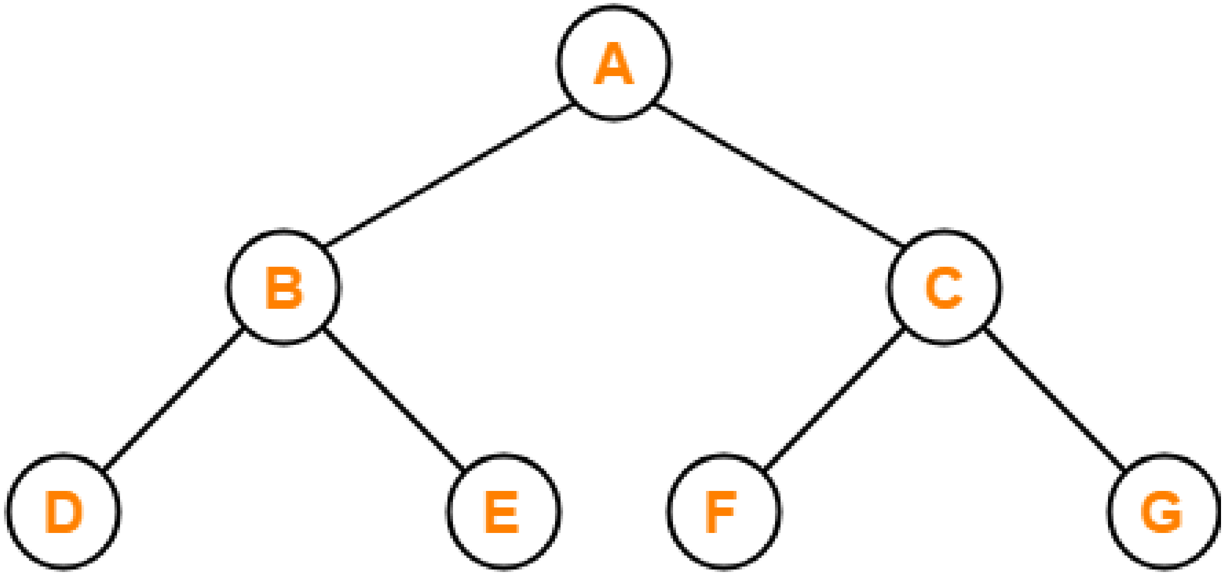

B
A
D
E
H
I
Right subtree
Left subtree
A B D E H I
A B D H I E
C
F
G
J
K
Left subtree
Right subtree
C F G J K
C F J G K

1. Preorder Traversal



A B D H I E C F J G K

1. Preorder Traversal

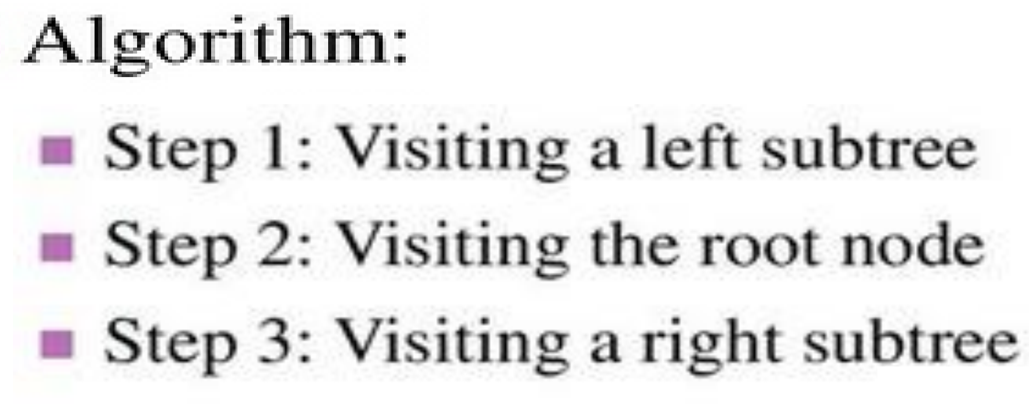


Preorder Traversal : A , B , D , E , C , F , G

Applications Of Preorder

- Preorder traversal is used to get prefix expression of an expression tree.
- Preorder traversal is used to create a copy of the tree.

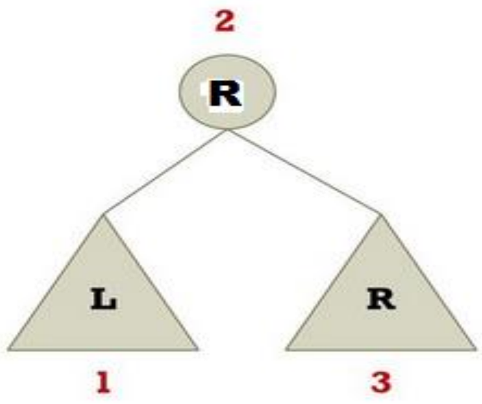
2. Inorder Traversal



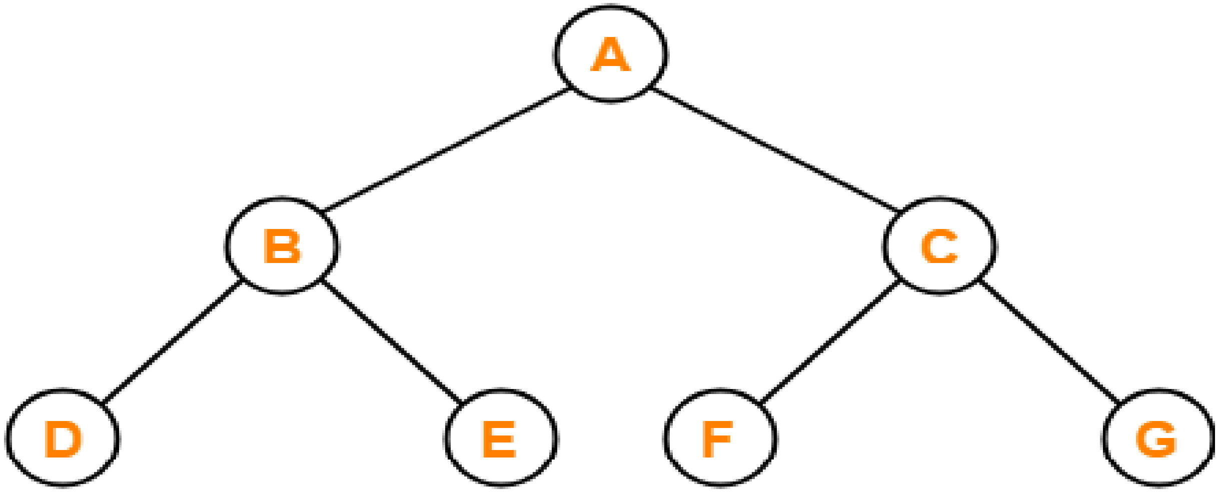
Algorithm:

- Step 1: Visiting a left subtree
- Step 2: Visiting the root node
- Step 3: Visiting a right subtree

Left → Root → Right



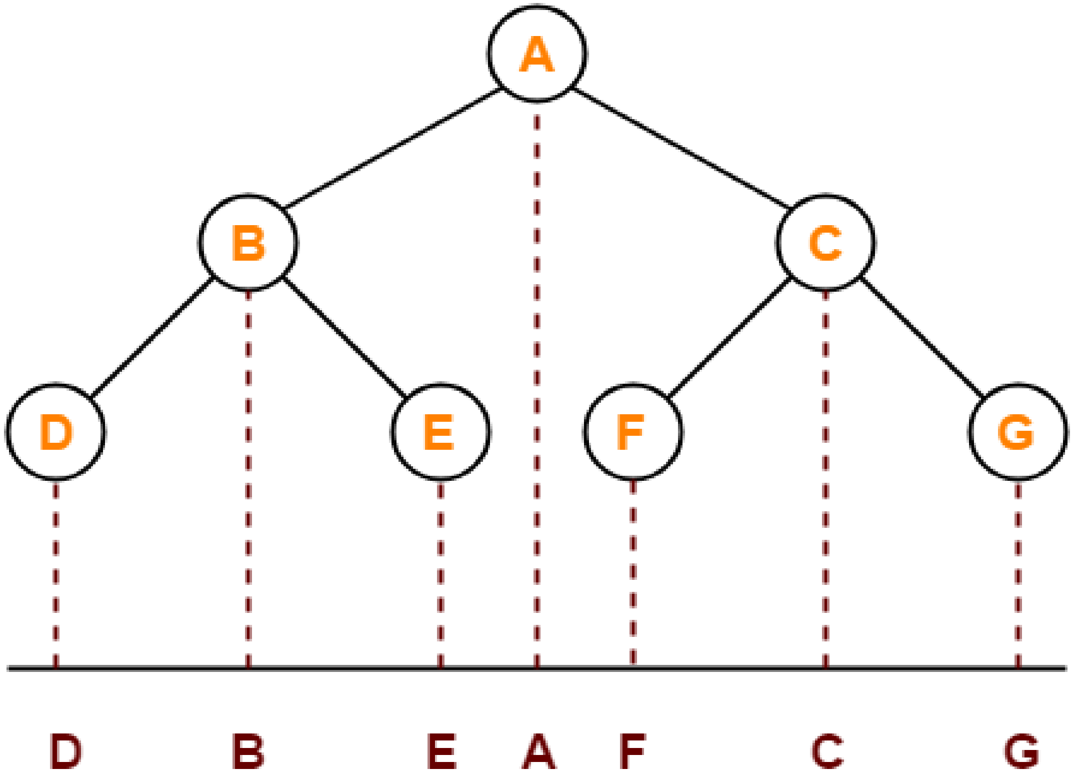
Consider the following example-



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

Inorder Traversal Shortcut

Keep a plane mirror horizontally at the bottom of the tree and take the projection of all the nodes.

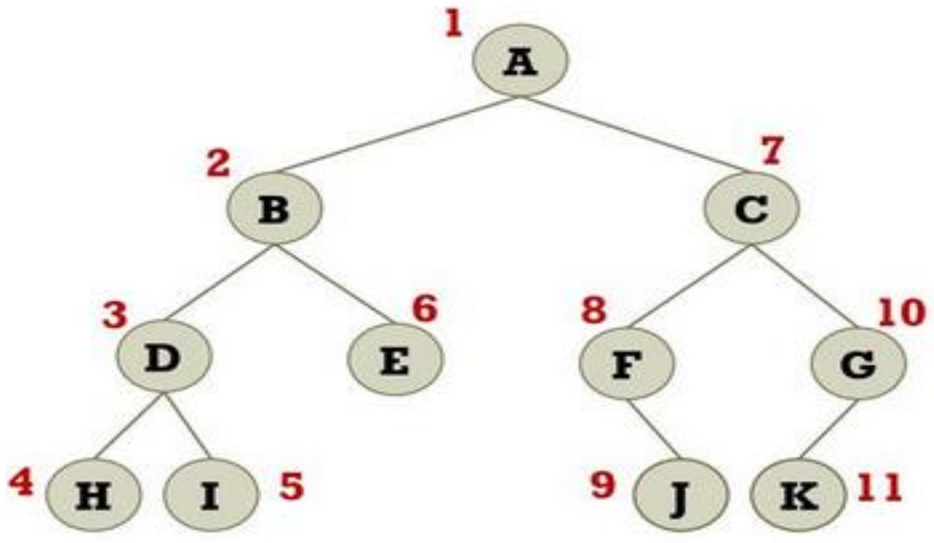


Inorder Traversal : D , B , E , A , F , C , G

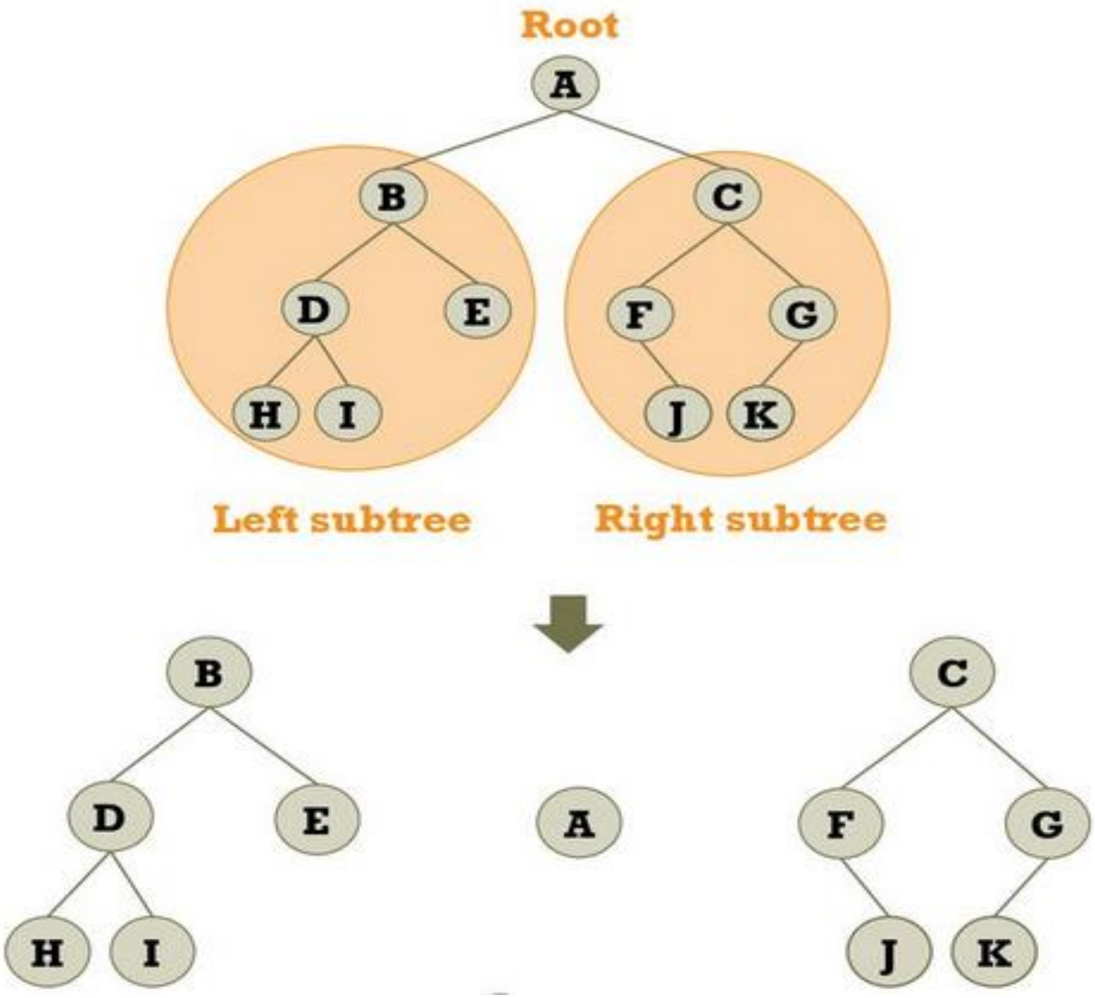
2. Inorder Traversal

Example

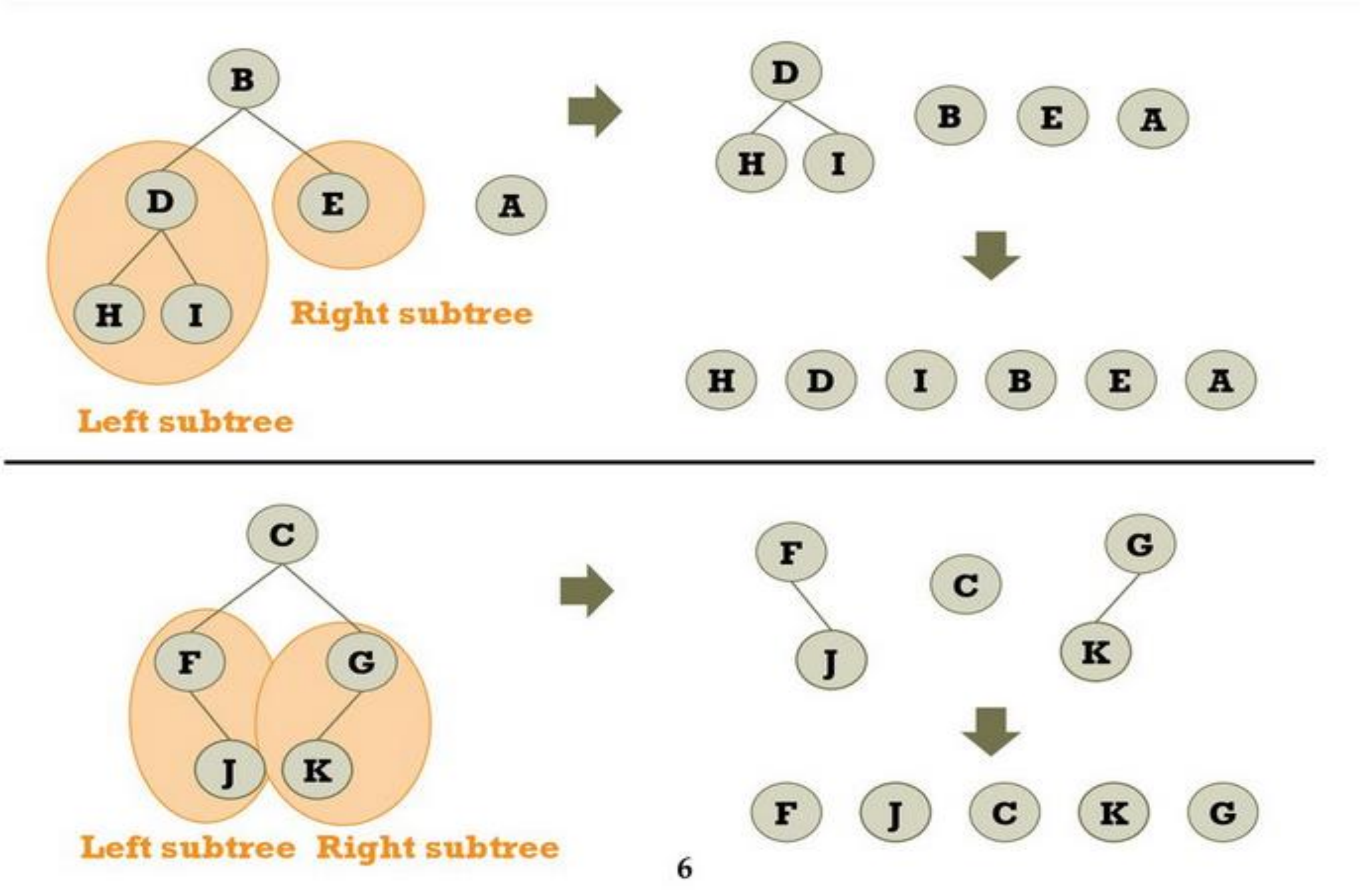
Consider the following example



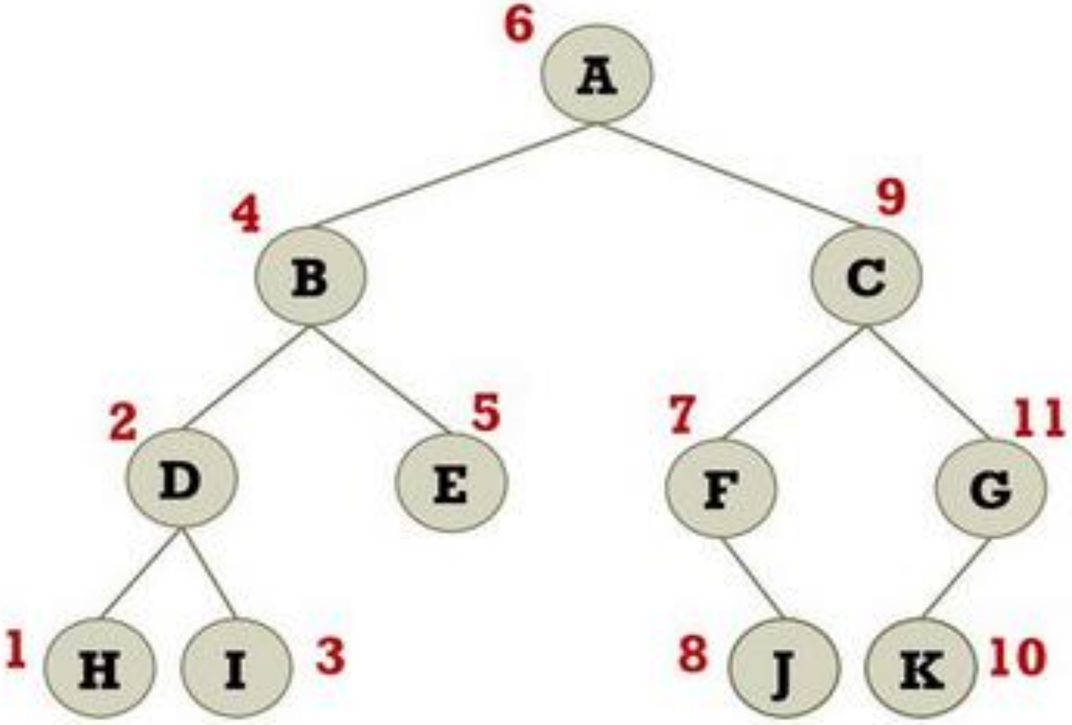
2. Inorder Traversal



2. Inorder Traversal



SO THE INORDER TRAVERSAL FOR GIVEN TREE IS

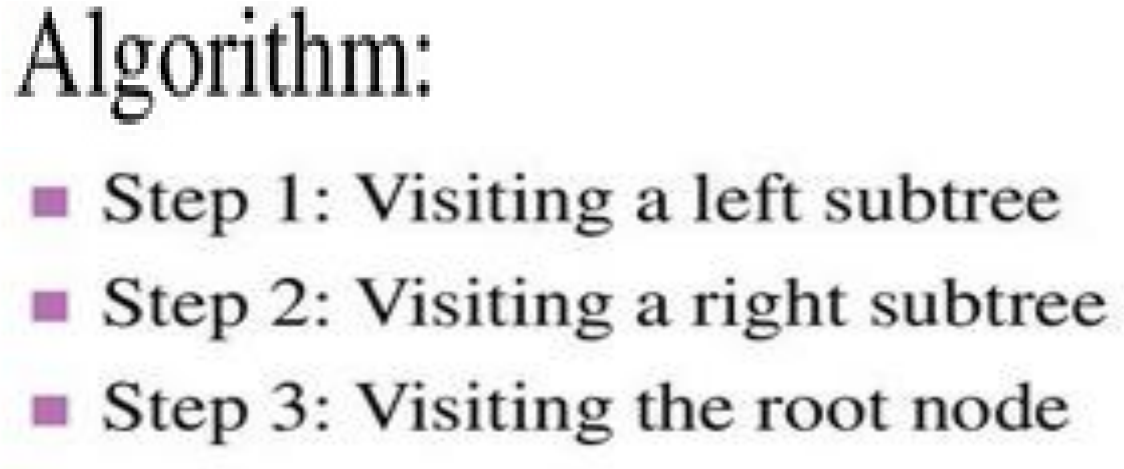


H D I B E A F J C K G

Applications Of Inorder

- Inorder traversal is used to get infix expression of an expression tree.

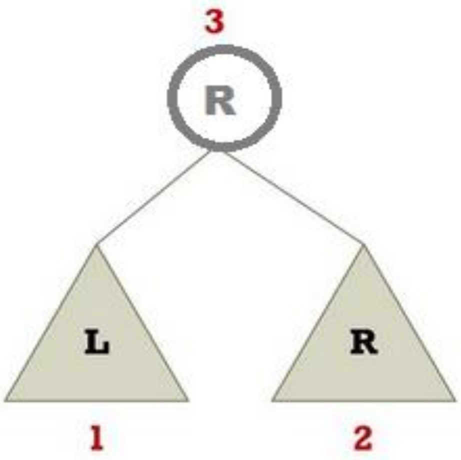
3. Postorder Traversal



Algorithm:

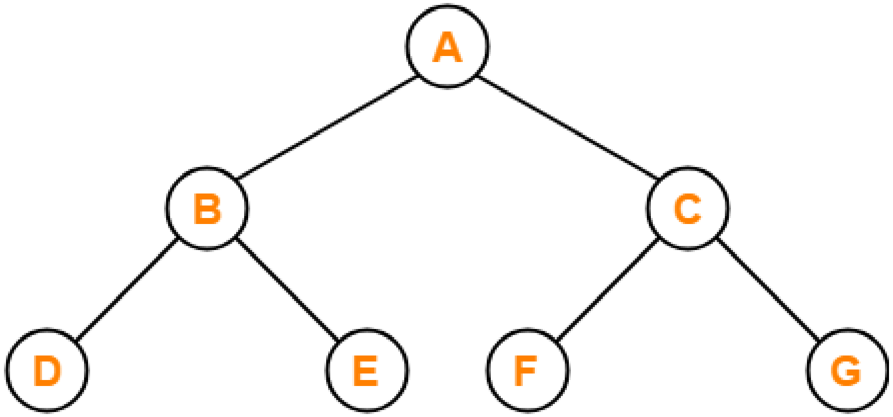
- Step 1: Visiting a left subtree
- Step 2: Visiting a right subtree
- Step 3: Visiting the root node

Left → Right → Root



3. Postorder Traversal

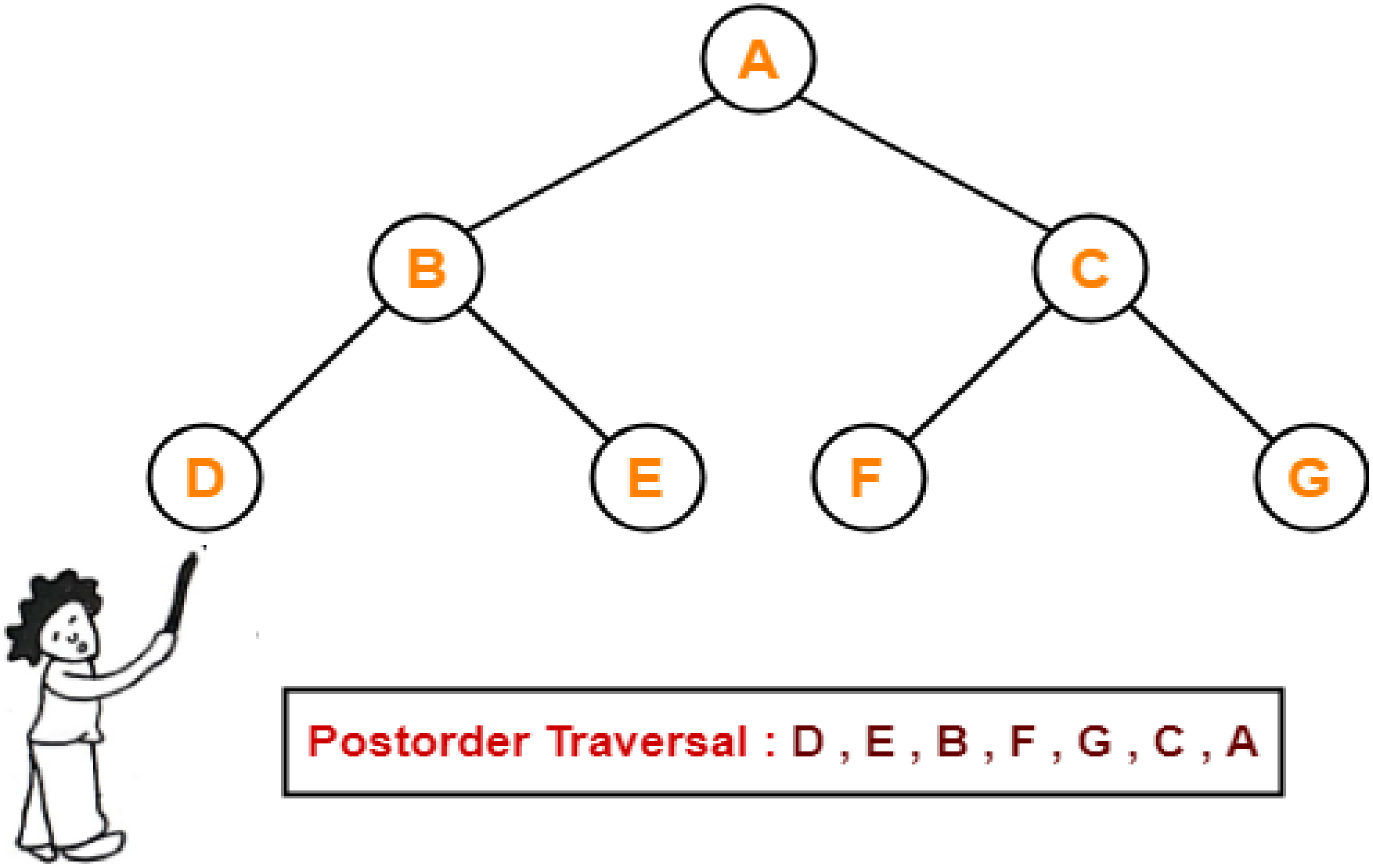
Consider the following example-



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

Postorder Traversal Shortcut

Pluck all the leftmost leaf nodes one by one.



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