15CSE374
INTRODUCTION TO DATA STRUCTURES AND ALGORITHMS

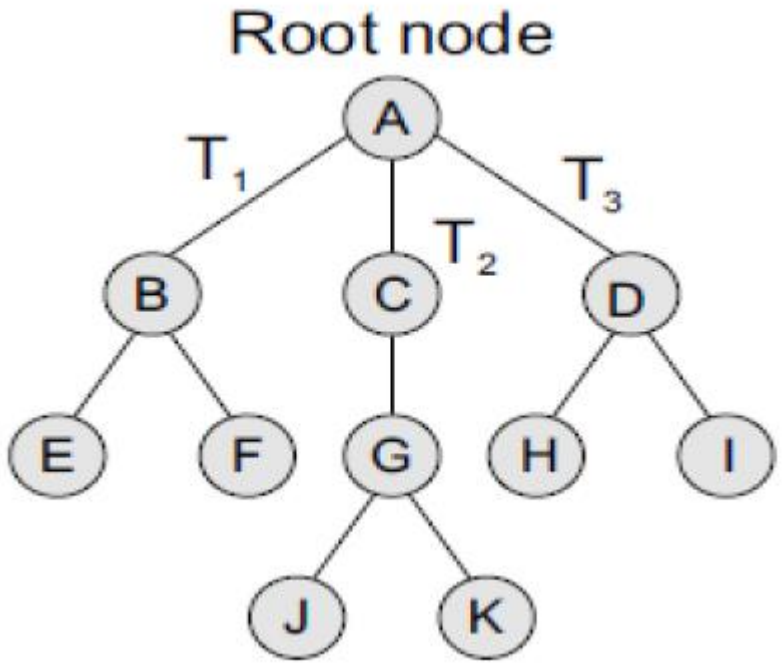
Sarath tv

Last Lecture

- NLDS.
- Tree – Basic Terminologies.
- Binary Tree
- Node Implementation of Binary Tree

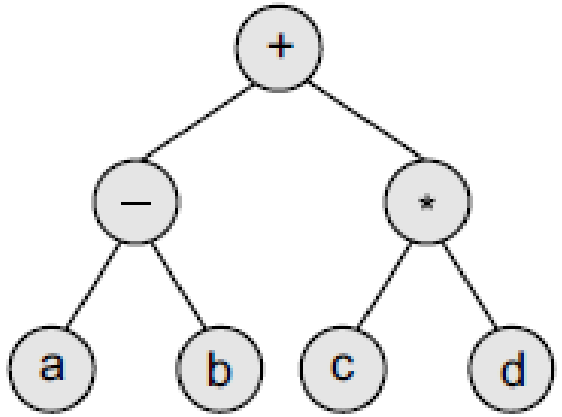
Sub Tree

- A **subtree** of a tree *T* is a tree consisting of a node in *T* and all of its descendants in *T*



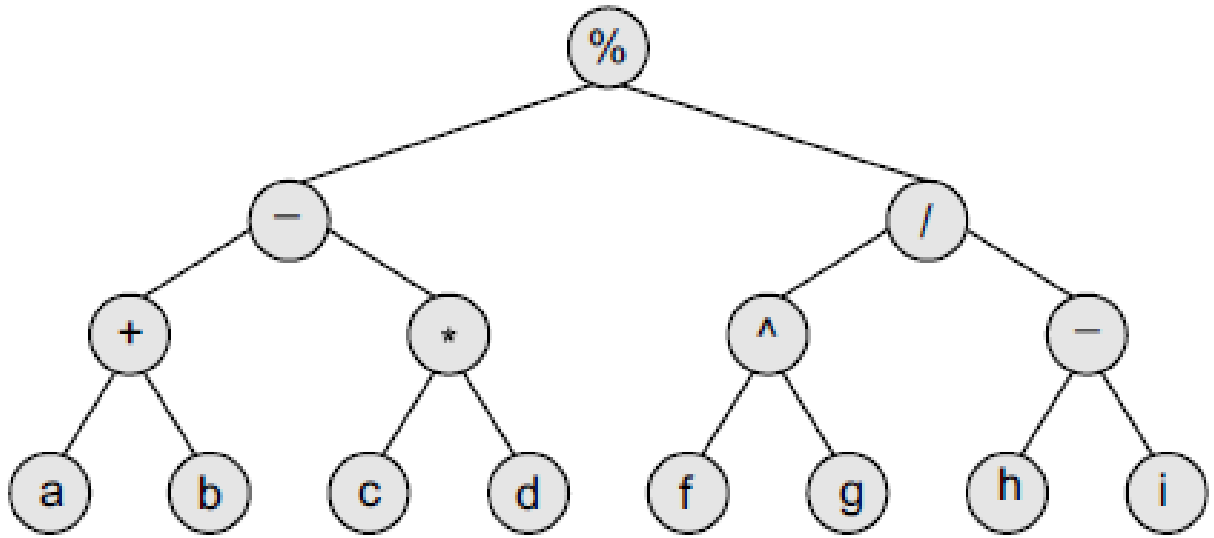
Expression Trees

Binary trees are widely used to store algebraic expressions. For example, consider Expression = (a – b) + (c * d)



Expression Trees

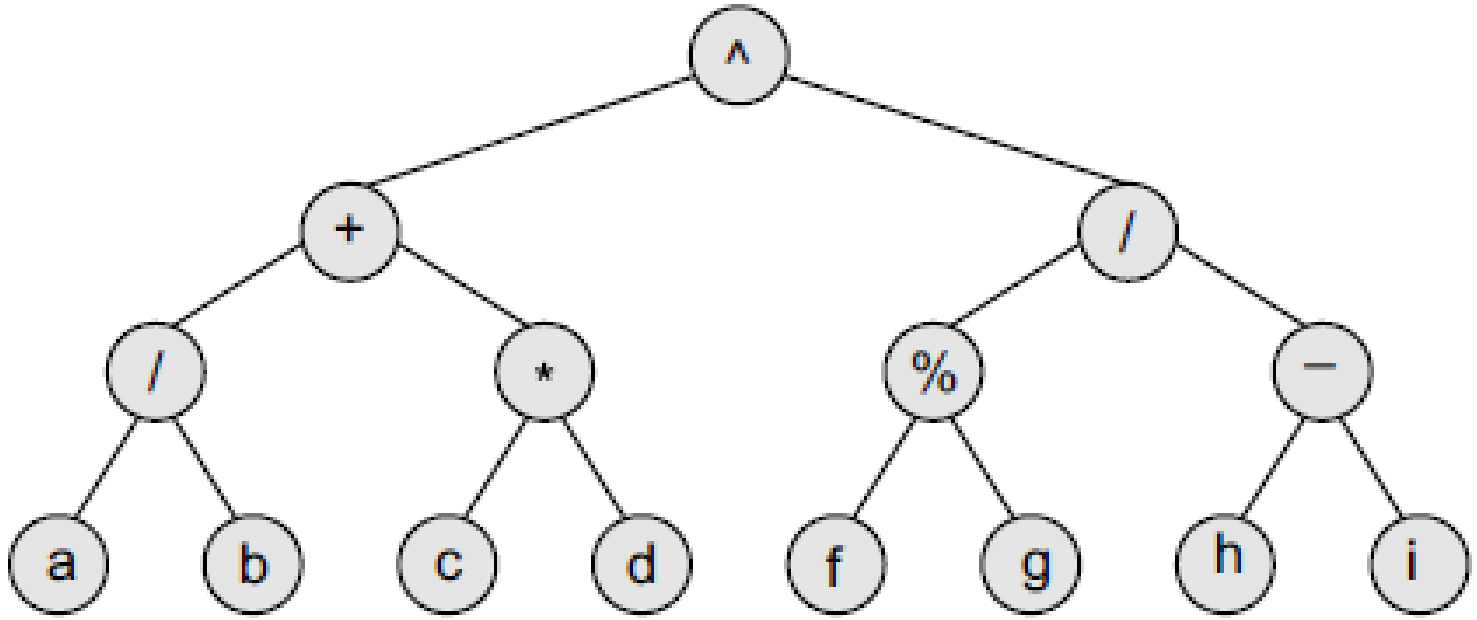
Given an expression, Exp = ((a + b) – (c * d)) % ((f ^g) / (h – i)), construct the corresponding binary tree.



Expression tree

Expression Trees

Given the binary tree, write down the expression that it represents




```
[{(a/b) + (c*d)} ^ {(f % g)/(h – i)}]
```

Trees Traversals

The process of visiting each node in the tree exactly once in a systematic way. Different algorithms for tree traversals differ in the order in which the nodes are visited.

Depth-First Search (DFS) : It starts with the root node and first visits all nodes of one branch as deep as possible of the chosen Node and before backtracking, it visits all other branches in a similar fashion.

Pre-order (NLR) Traversal

In-order (LNR) Traversal

Post-order (LRN) Traversal

Breadth-First Search (BFS) Algorithm: It also starts from the root node and visits all nodes of current depth before moving to the next depth in the tree.

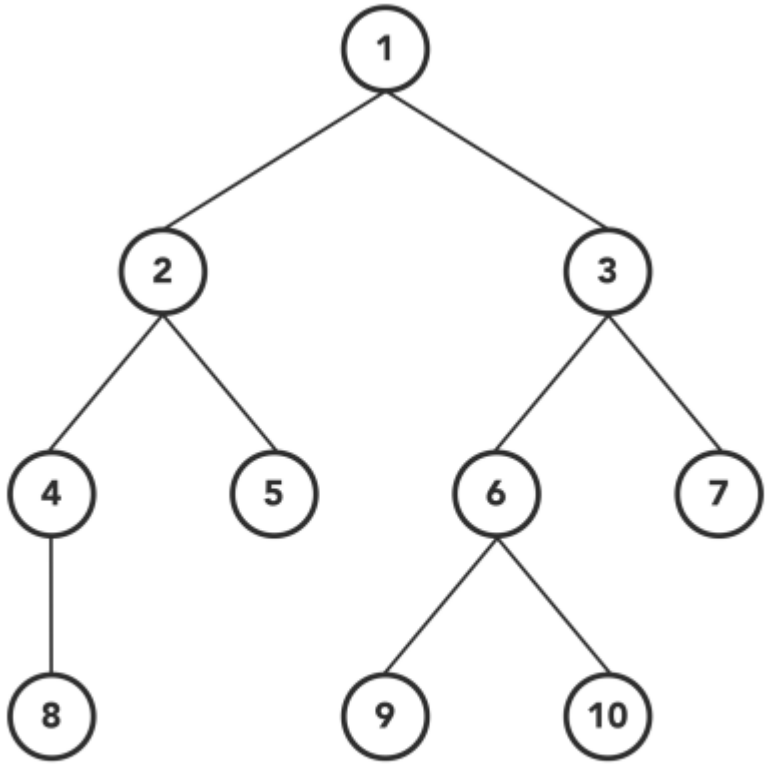
Level-order Traversal

Pre-order Traversal

The following operations are performed recursively at each node:

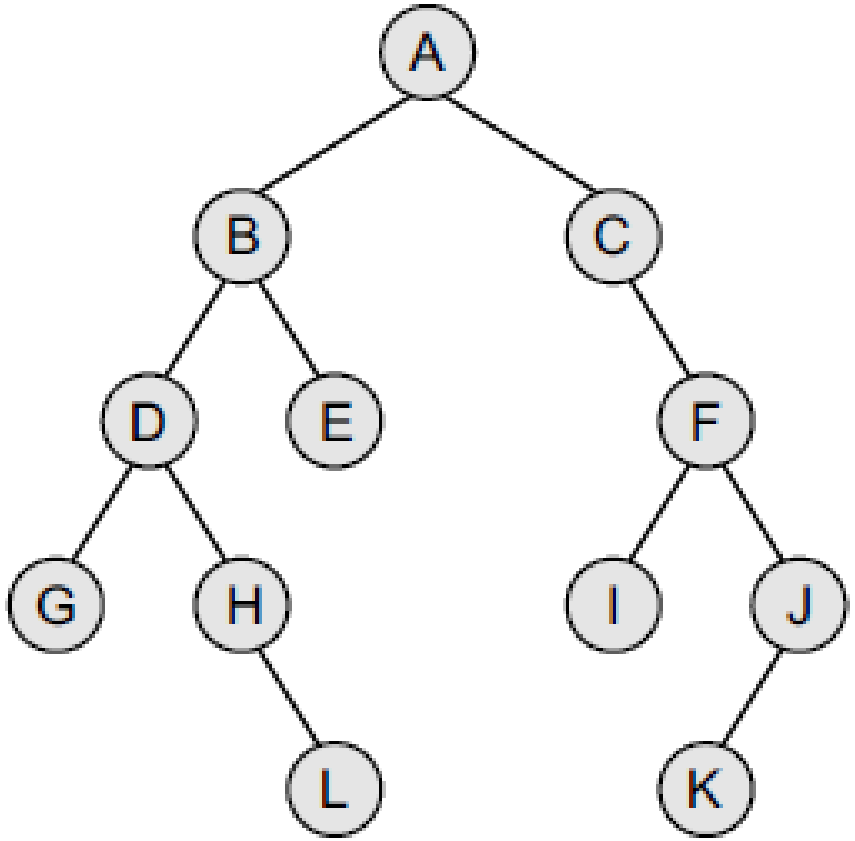
1. Visiting the root node,
2. Traversing the left sub-tree, and finally
3. Traversing the right sub-tree.

NLR

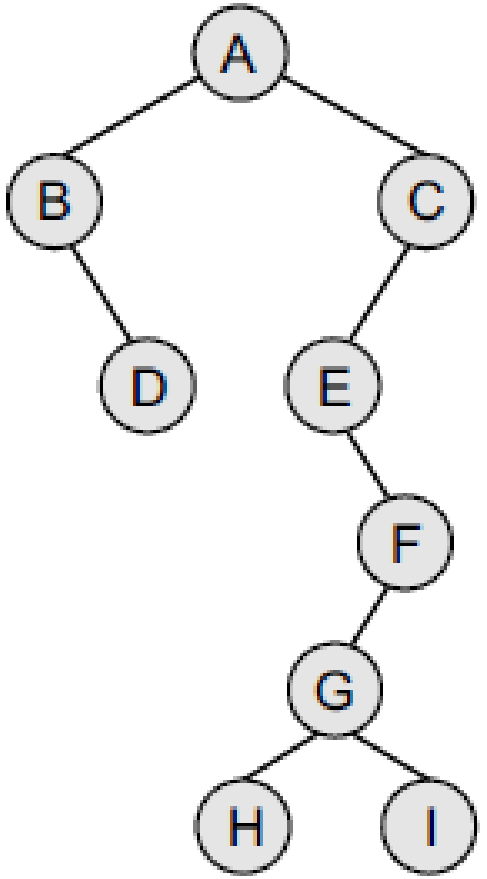


Pre-order Traversal

NLR



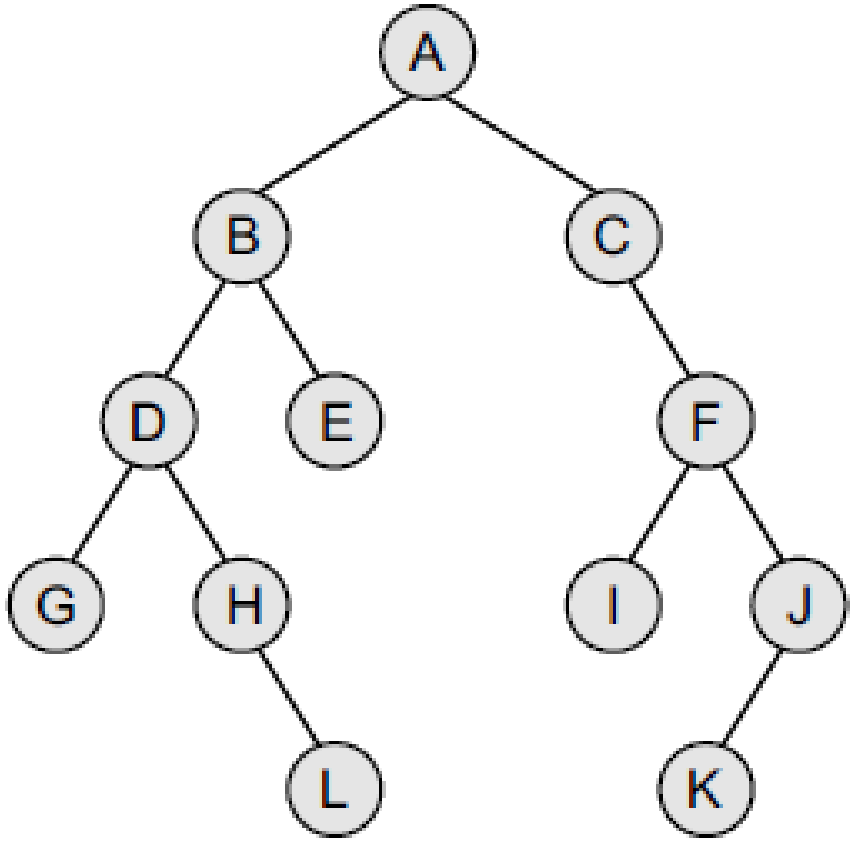
(a)



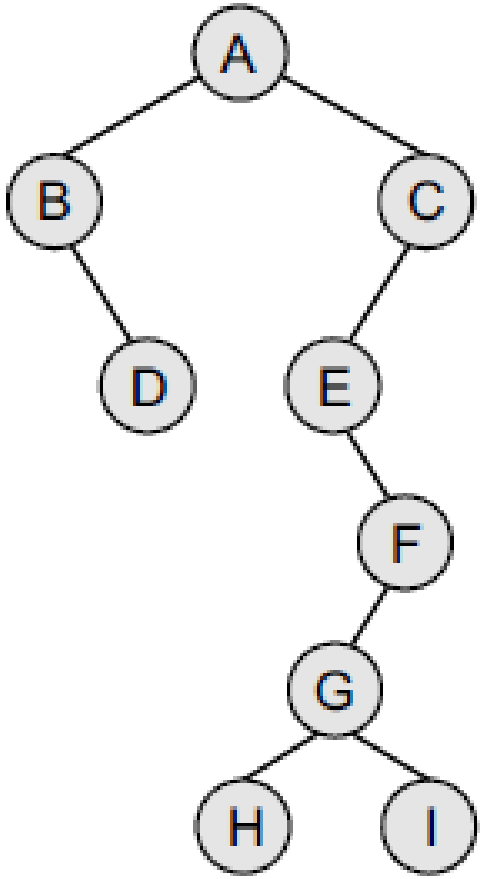
(b)

Pre-order Traversal

NLR



(a)



(b)

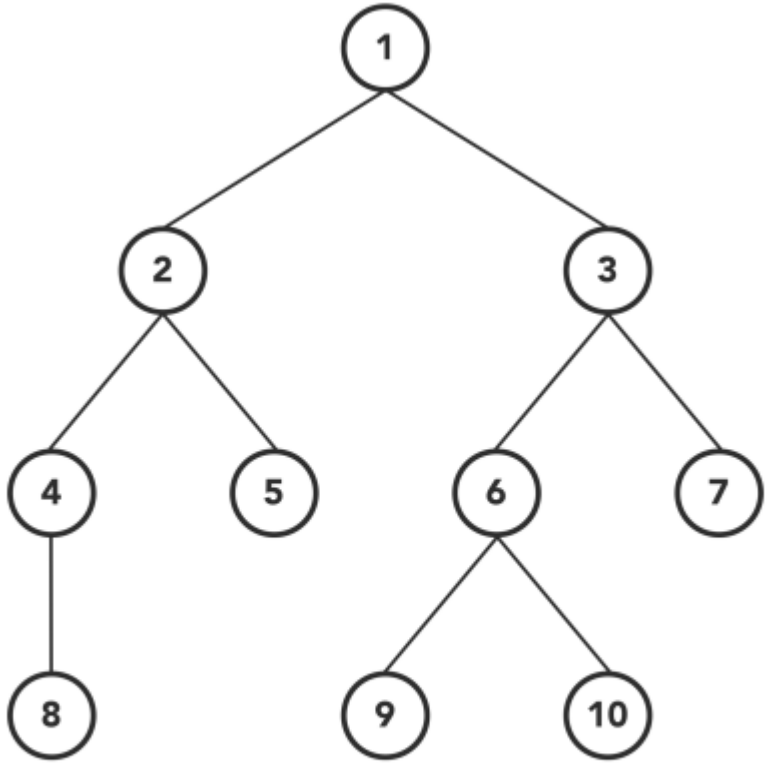
(a) A, B, D, G, H, L, E, C, F, I, J, K, (b) A, B, D, C, E, F, G, H, I

In-order Traversal

The following operations are performed recursively at each node:

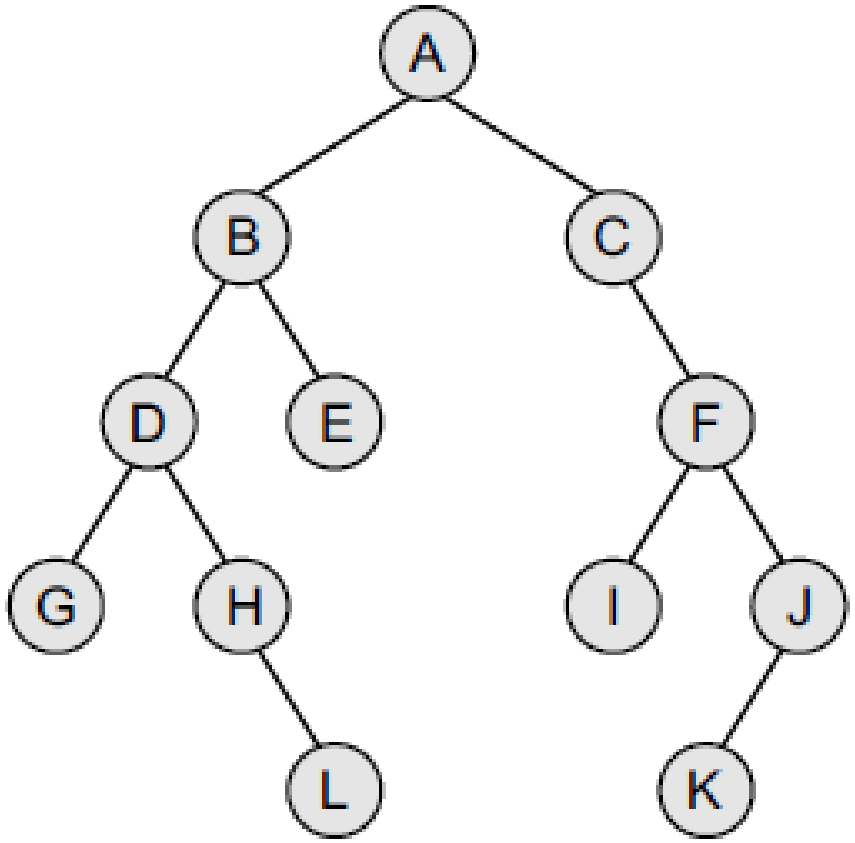
1. Traversing the left sub-tree,
2. Visiting the root node, and finally
3. Traversing the right sub-tree.

LNR

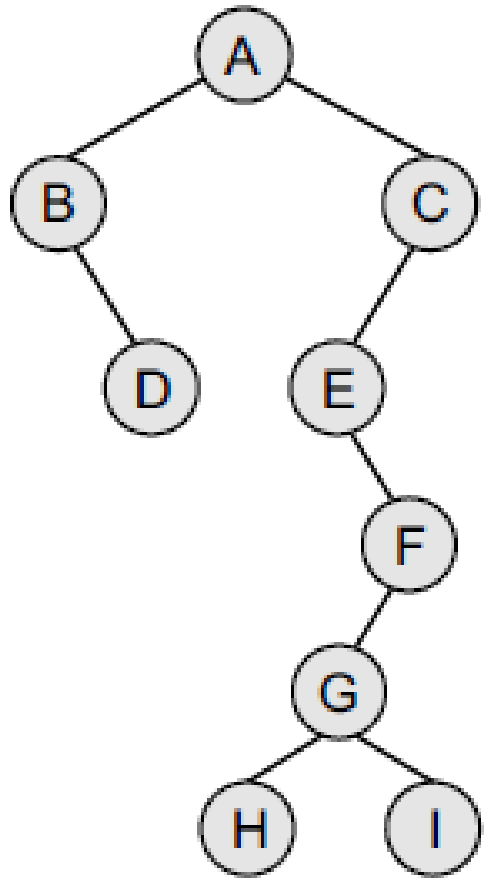


In-order Traversal

LNR



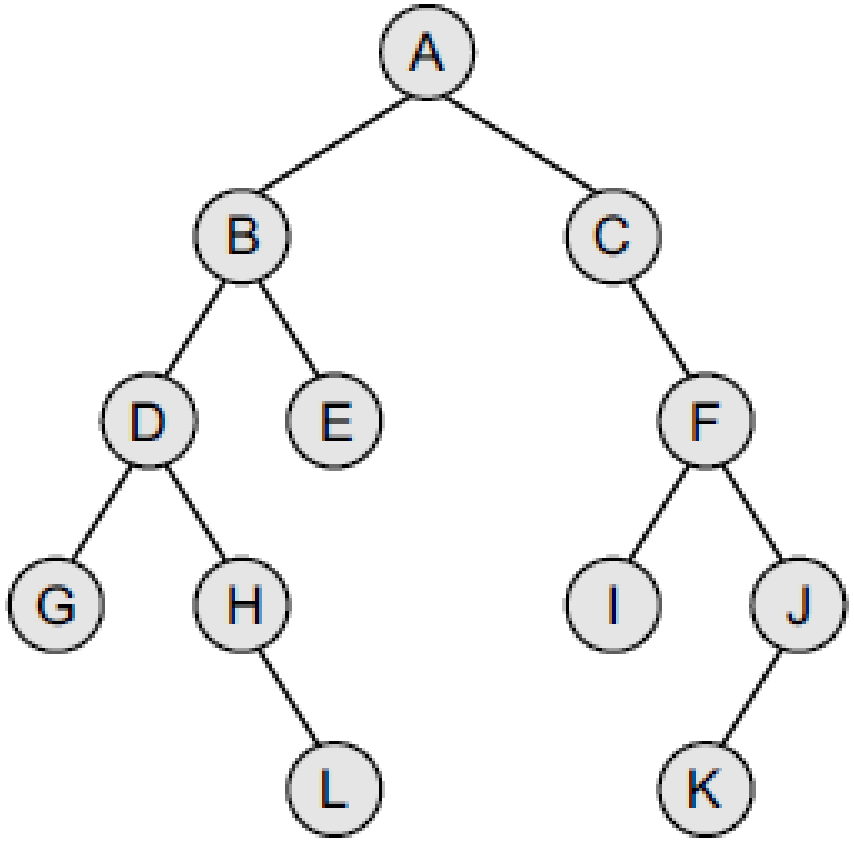
(a)



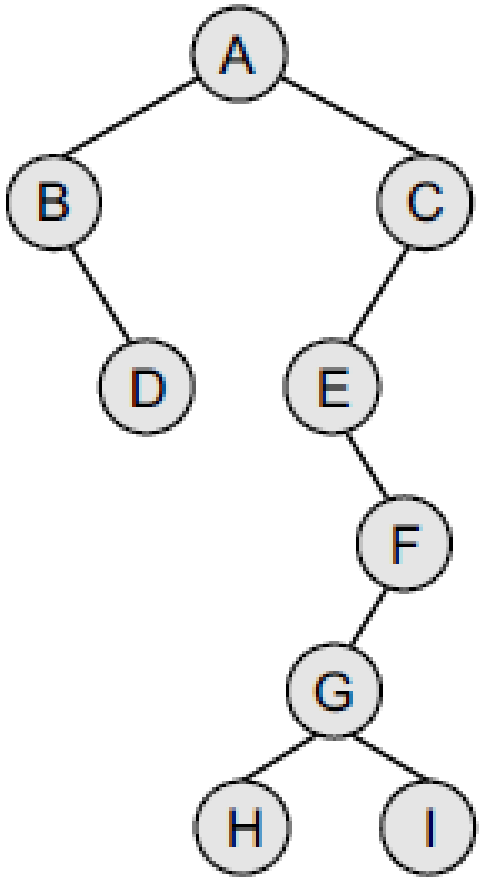
(b)

In-order Traversal

LNR



(a)



(b)

(a) G, D, H, L, B, E, A, C, I, F, K, J

(b) B, D, A, E, H, G, I, F, C

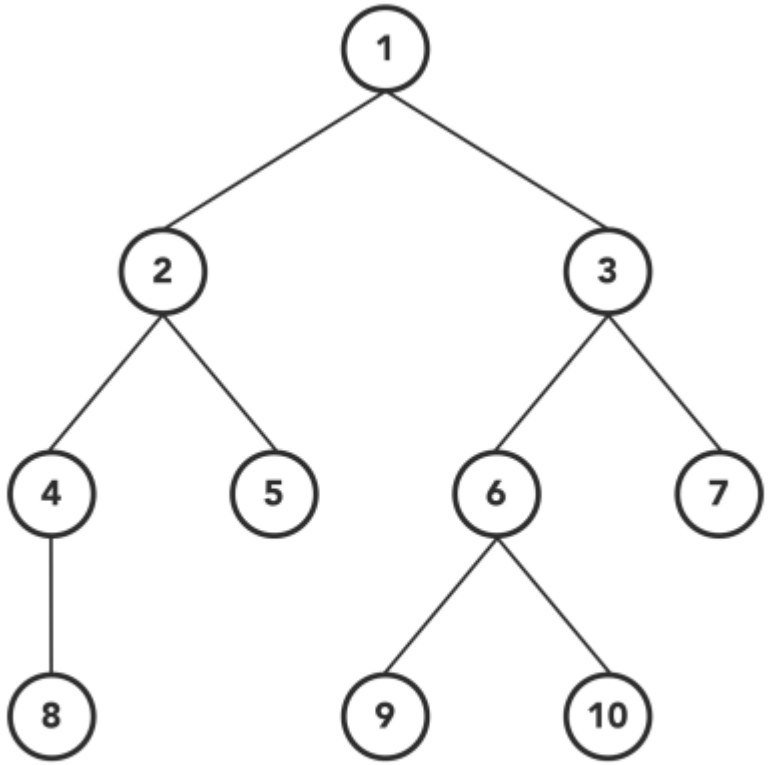
Post-order Traversal

The following operations are performed recursively at each node:

1. Traversing the left sub-tree,
2. Traversing the right sub-tree, and finally
3. Visiting the root node.

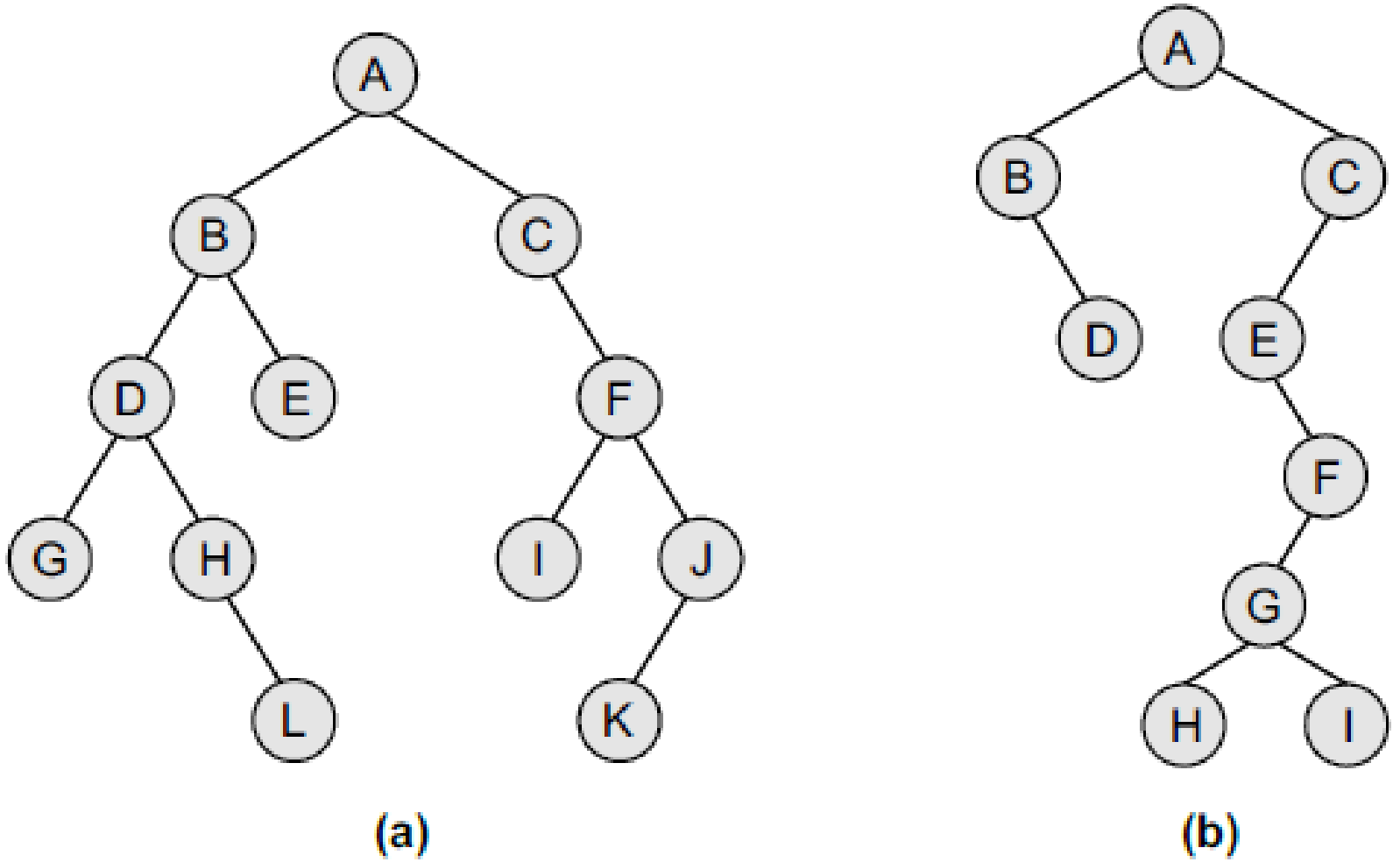
Post-order Traversal

LRN



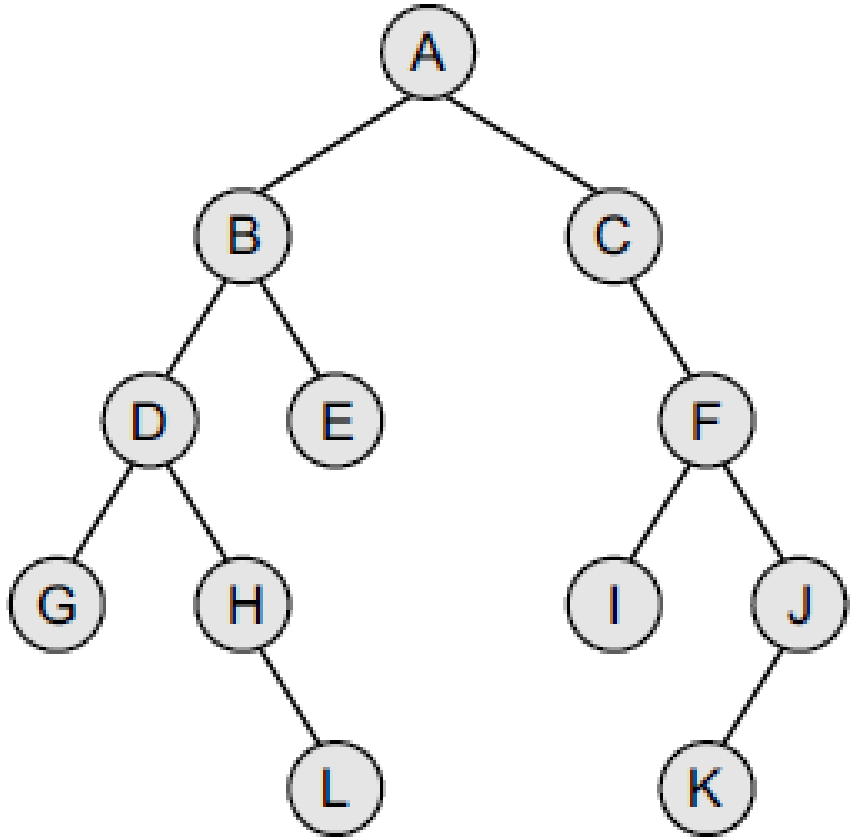
Post-order Traversal

LRN

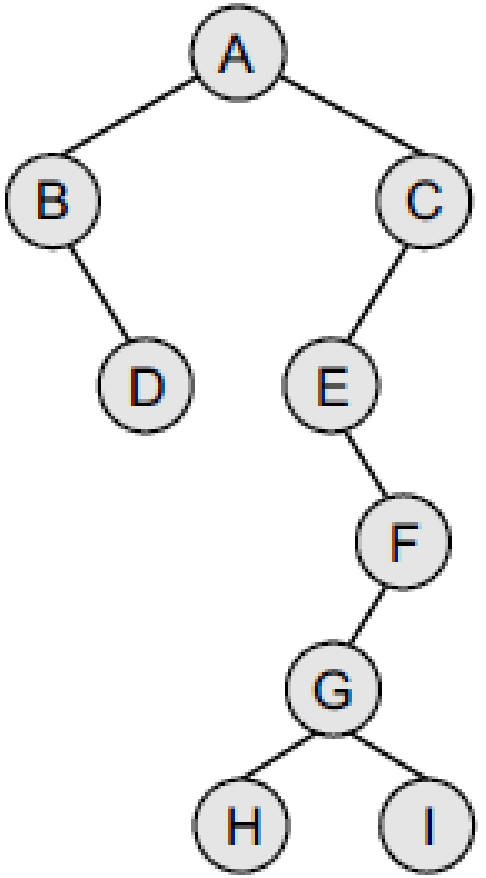


Post-order Traversal

LRN



(a)



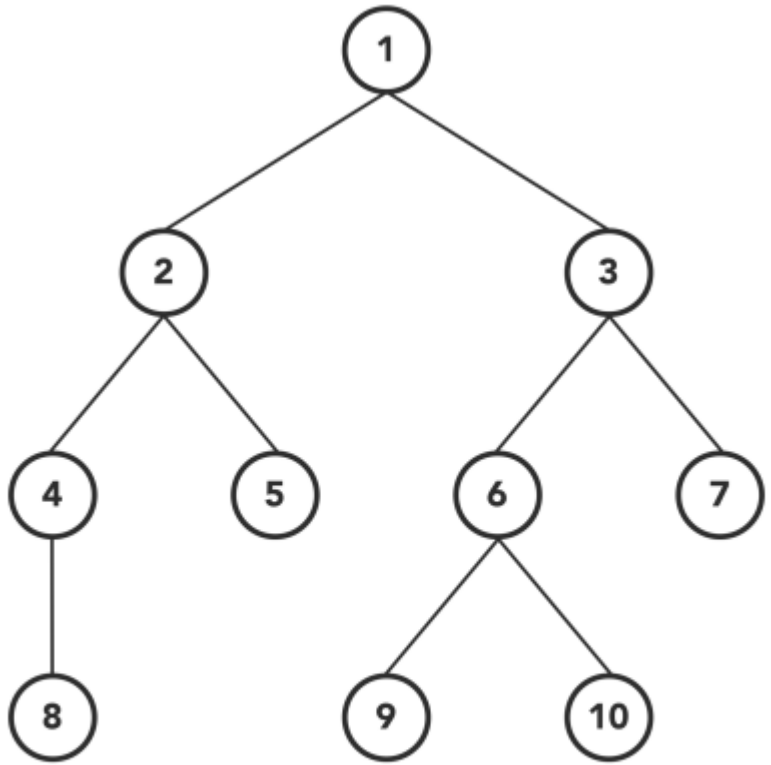
(b)

(a) G, L, H, D, E, B, I, K, J, F, C, A

(b) D, B, H, I, G, F, E, C, A

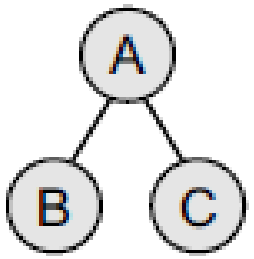
Level-order (breadth-first) Traversal

All the nodes at a level are accessed before going to the next level.

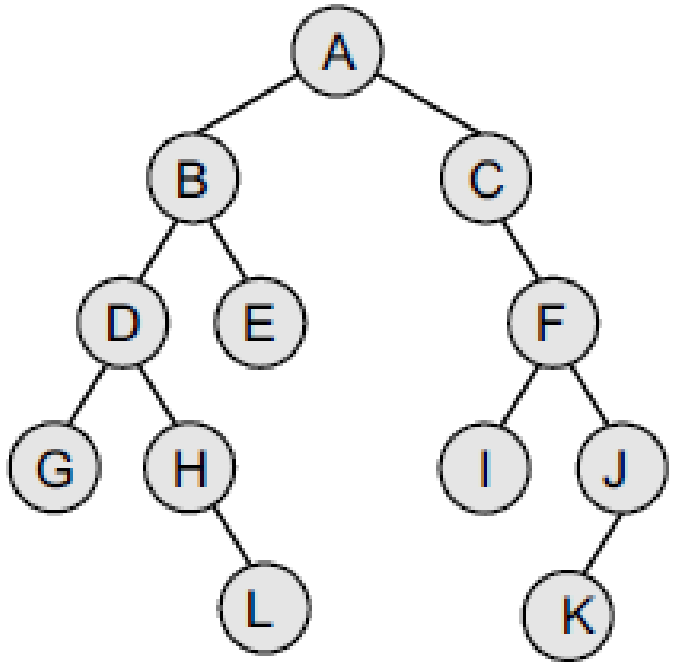


Level-order (breadth-first) Traversal

All the nodes at a level are accessed before going to the next level.

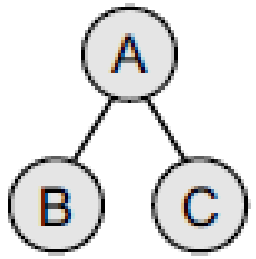


(a)



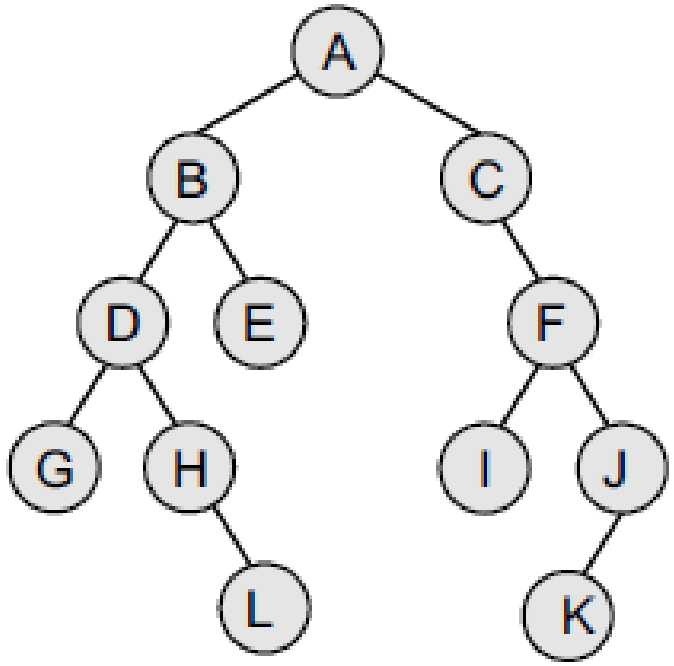
Level-order (breadth-first) Traversal

All the nodes at a level are accessed before going to the next level.



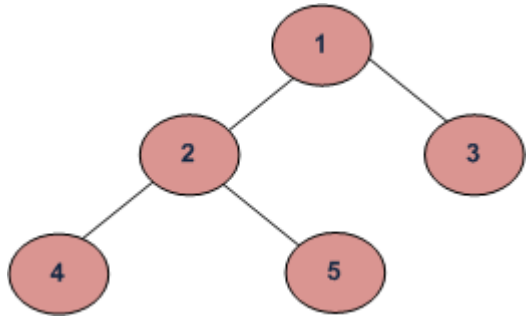
(a)

TRAVERSAL ORDER:
A, B, and C



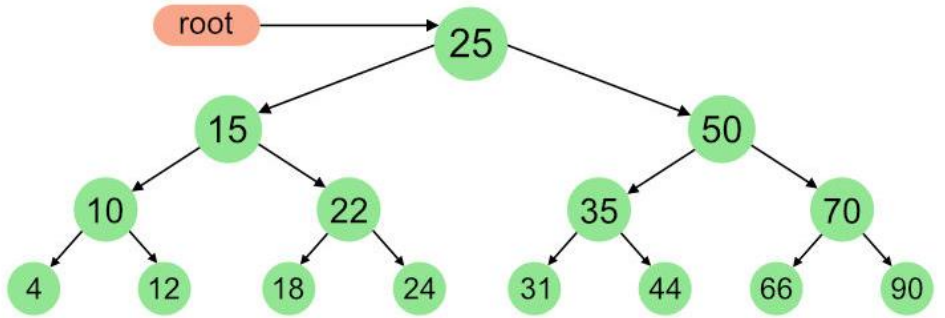
A, B, C, D, E, F, G, H, I, J, L, K

Depth First Traversals:
(a) Inorder (LNR) :
(b) Preorder (NLR)
(c) Postorder (LRN)



Depth First Traversals:
(a) Inorder (Left, Root, Right) : 4 2 5 1 3
(b) Preorder (Root, Left, Right) : 1 2 4 5 3
(c) Postorder (Left, Right, Root) : 4 5 2 3 1
Breadth First or Level Order Traversal : 1 2 3 4 5

Depth First Traversals:
(a) Inorder (LNR) :
(b) Preorder (NLR)
(c) Postorder (LRN)

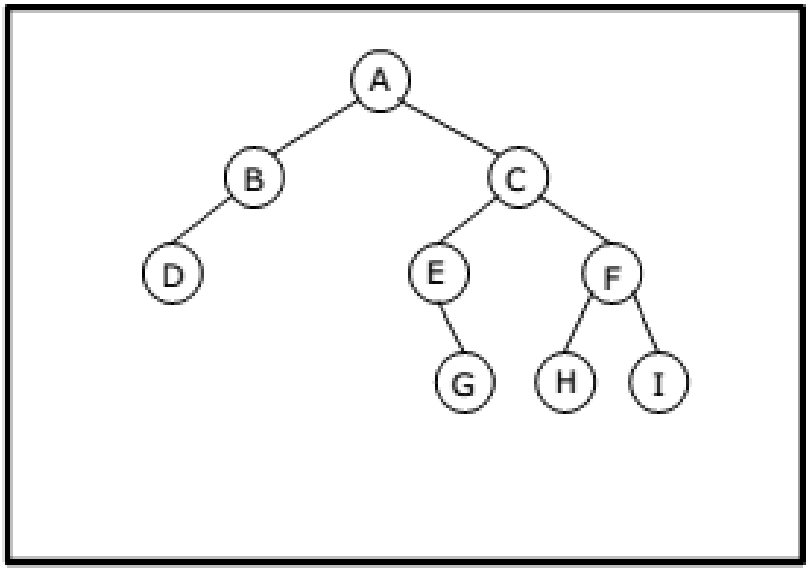


InOrder(root) visits nodes in the following order:
4, 10, 12, 15, 18, 22, 24, 25, 31, 35, 44, 50, 66, 70, 90

A Pre-order traversal visits nodes in the following order:
25, 15, 10, 4, 12, 22, 18, 24, 50, 35, 31, 44, 70, 66, 90

A Post-order traversal visits nodes in the following order:
4, 12, 10, 18, 24, 22, 15, 31, 44, 35, 66, 90, 70, 50, 25

Depth First Traversals:
(a) Inorder (LNR) :
(b) Preorder (NLR)
(c) Postorder (LRN)

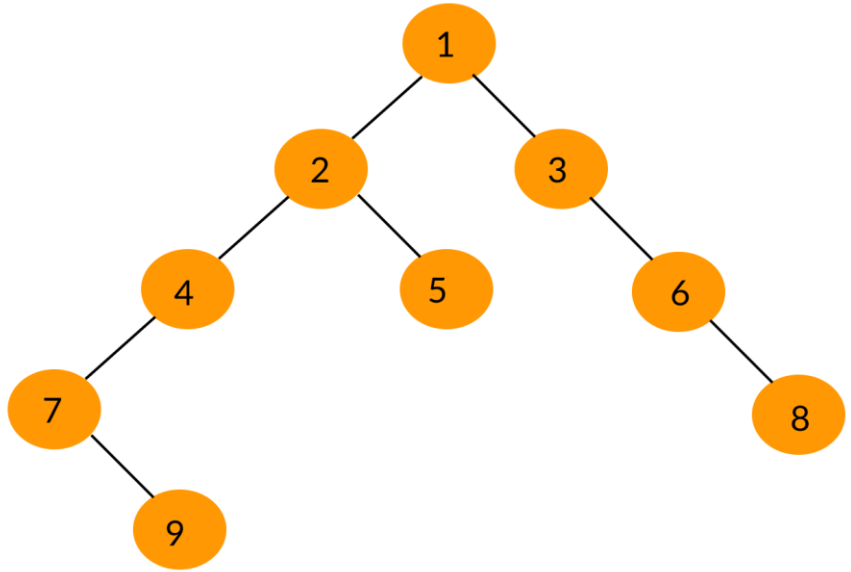


Binary Tree

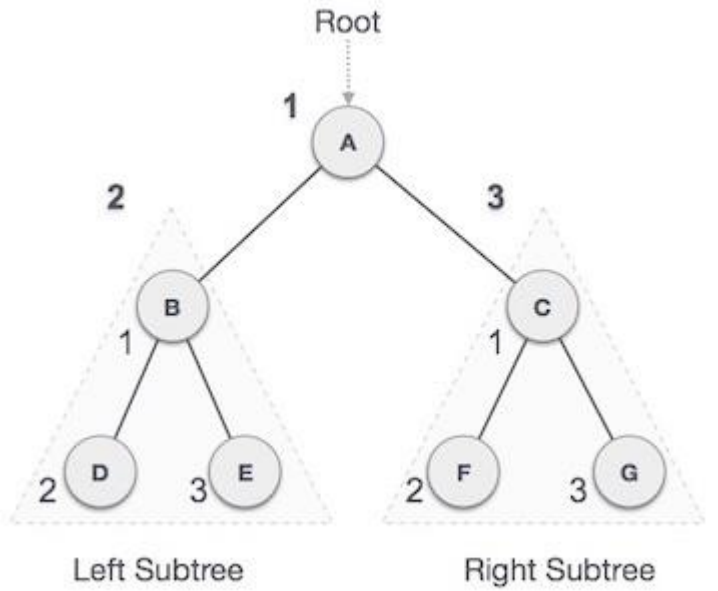
- Preorder traversal yields:
  A, B, D, C, E, G, F, H, I
- Postorder traversal yields:
  D, B, G, E, H, I, F, C, A
- Inorder traversal yields:
  D, B, A, E, G, C, H, F, I
- Level order traversal yields:
  A, B, C, D, E, F, G, H, I

Pre, Post, Inorder and level order Traversing

Depth First Traversals:
(a) Inorder (LNR) :
(b) Preorder (NLR)
(c) Postorder (LRN)



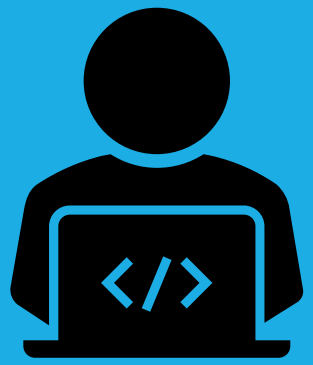
Inorder Traversal: 7 9 4 2 5 1 3 6 8
Preorder Traversal: 1 2 4 7 9 5 3 6 8
Postorder Traversal: 9 7 4 5 2 8 6 3 1



1 D → B → E → A → F → C → G

2 A → B → D → E → C → F → G

3 D → E → B → F → G → C → A



THANK YOU!!!!!

- Data structures Introduction to Trees

https://www.youtube.com/watch?v=qH6yxkw0u78

- Find height of a binary tree

https://www.youtube.com/watch?v=_pnqMz5nrRs&t=1s

- Data structures: Binary Tree

https://www.youtube.com/watch?v=H5JubkIy_p8&t=8s

References

Binary tree traversal Preorder, Inorder, Postorder

https://www.youtube.com/watch?v=gm8DUJJhmY4&t=4s

Binary tree Level Order Traversal

https://www.youtube.com/watch?v=86g8jAQug04&t=7s

Binary tree traversal - breadth-first and depth-first strategies

https://www.youtube.com/watch?v=9RHO6jU--GU&t=3s