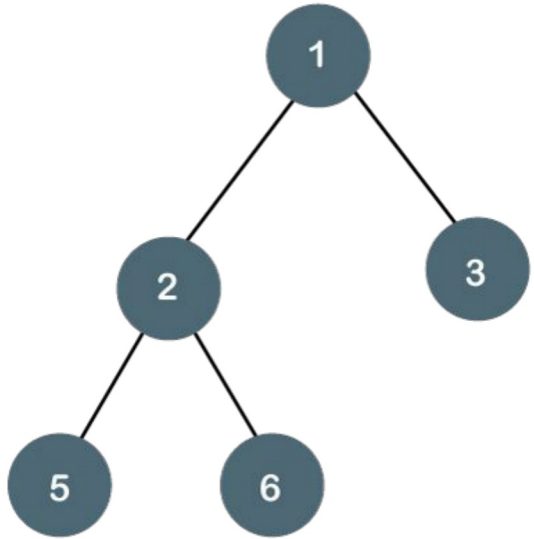
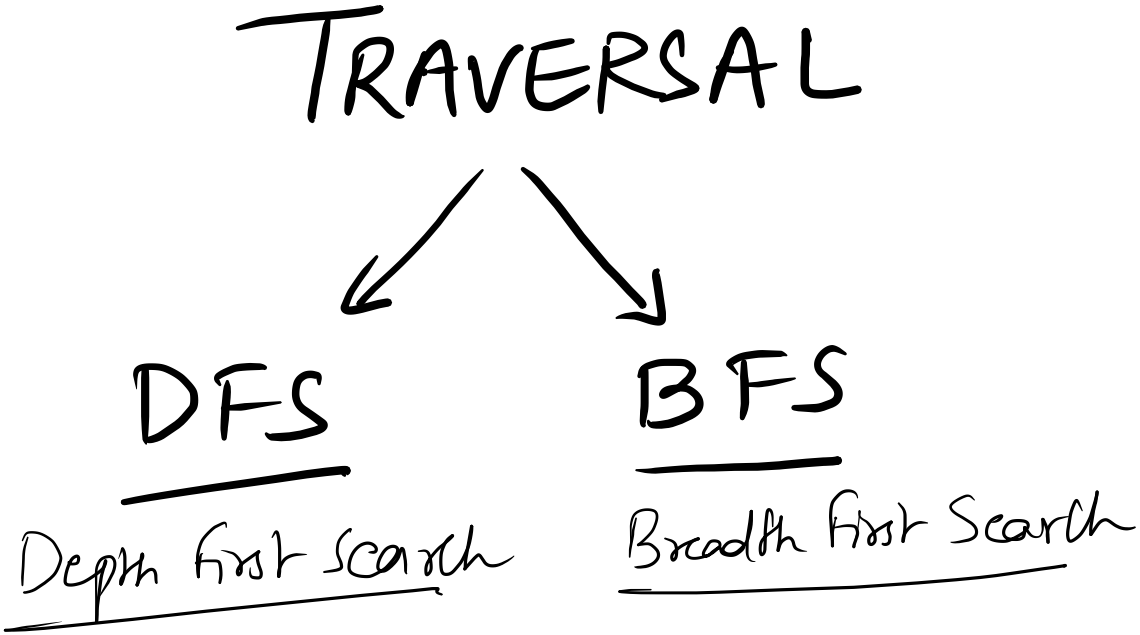
Traversal Algorithms
TREES

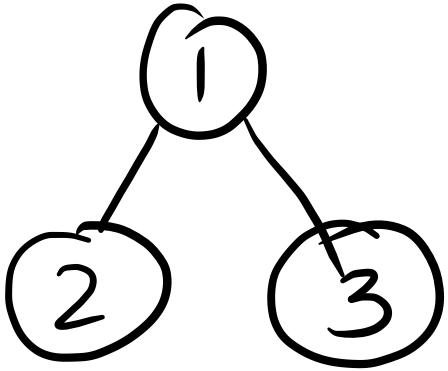
By Gladden Rumao




TRAVERSAL
DFS
Depth First Search
BFS
Breadth First Search

DFS

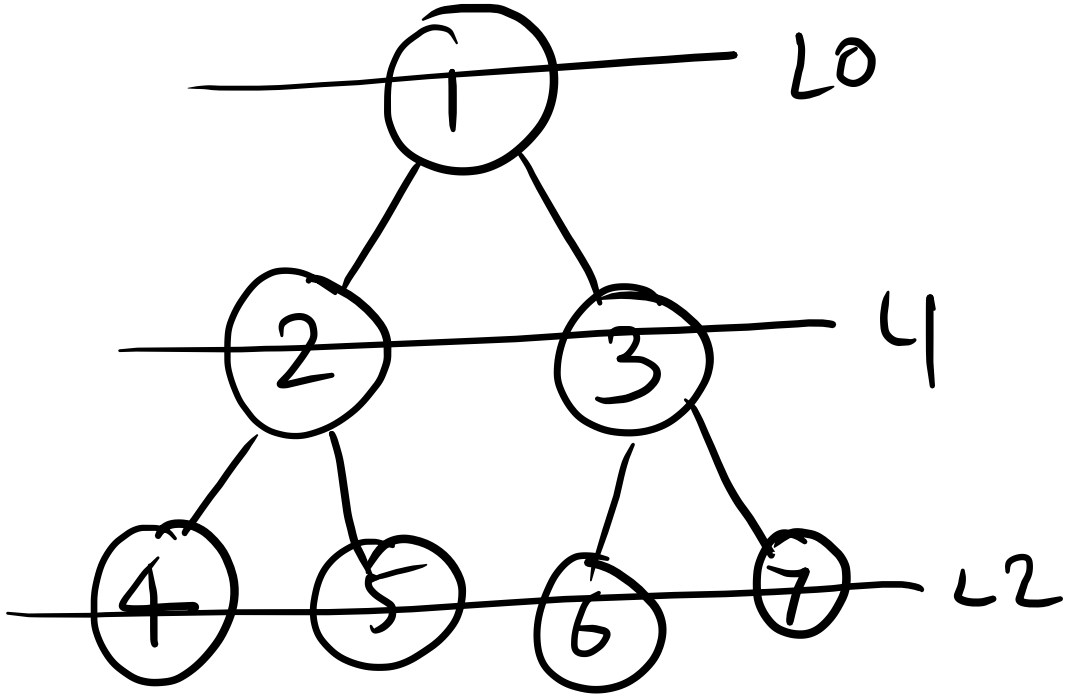
1. Inorder
2. Preorder
3. Postorder



2 1 3

1 2 3

2 3 1



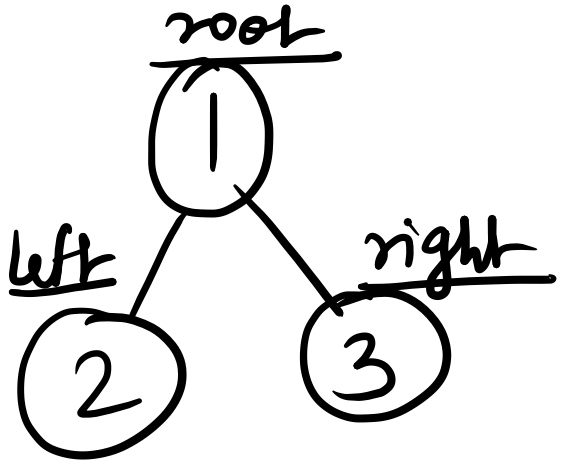
depth

breath

Inorder

left root right

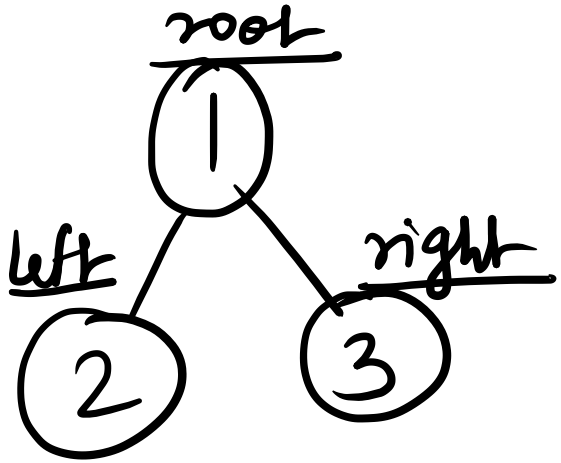
2 1 3



Preorder

root left right

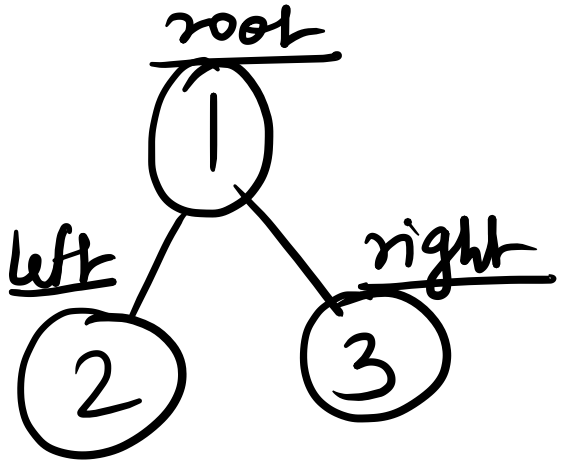
1 2 3

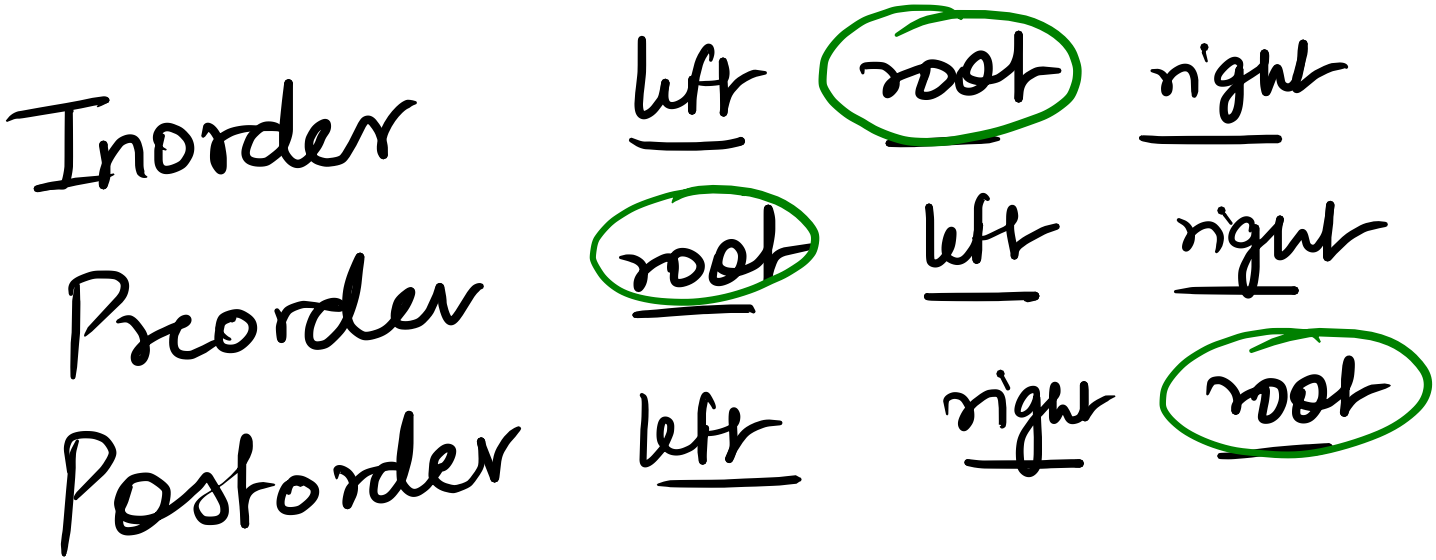


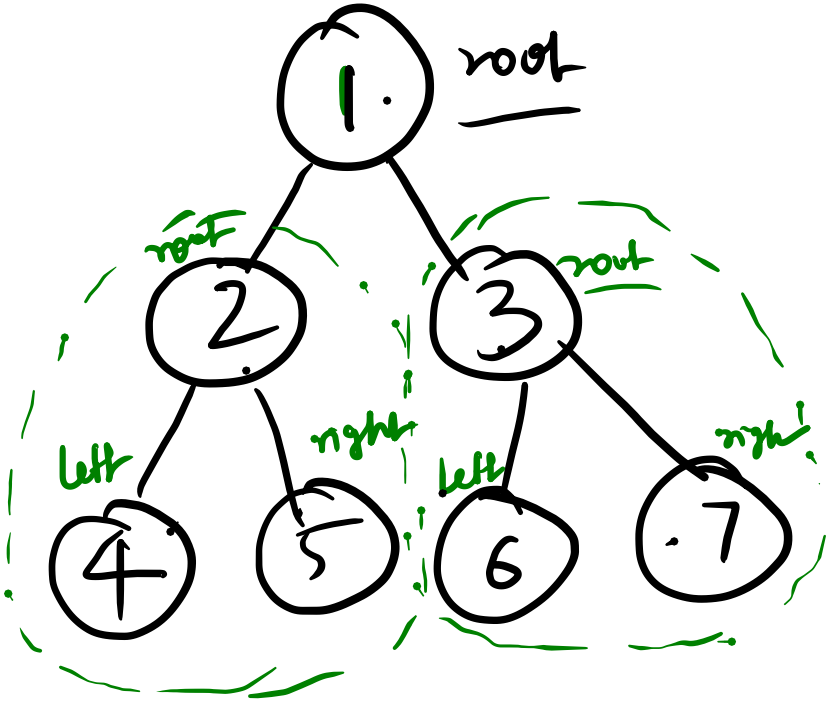
Postorder

left right root

2 3 1



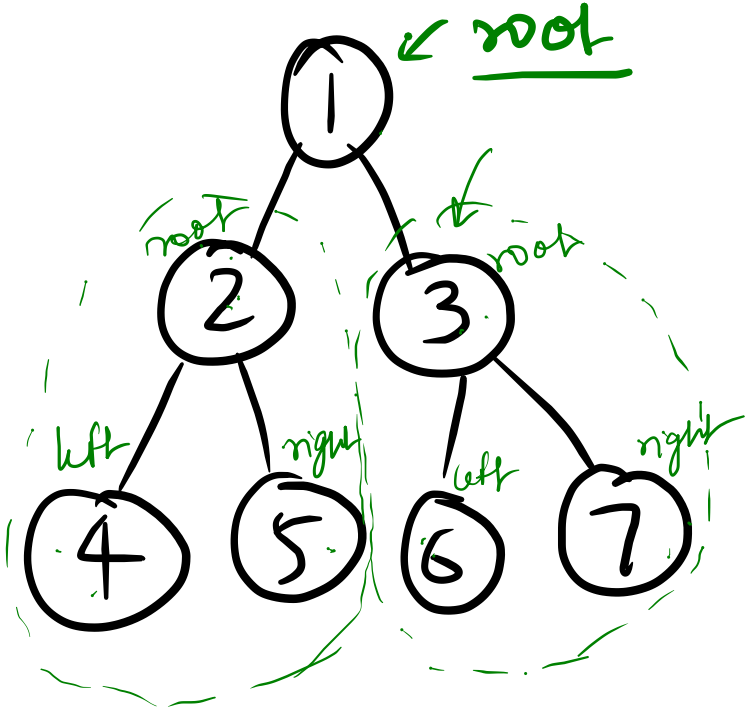

Inorder
left
root
right
Preorder
root
left
right
Postorder
left
right
root



Inorder

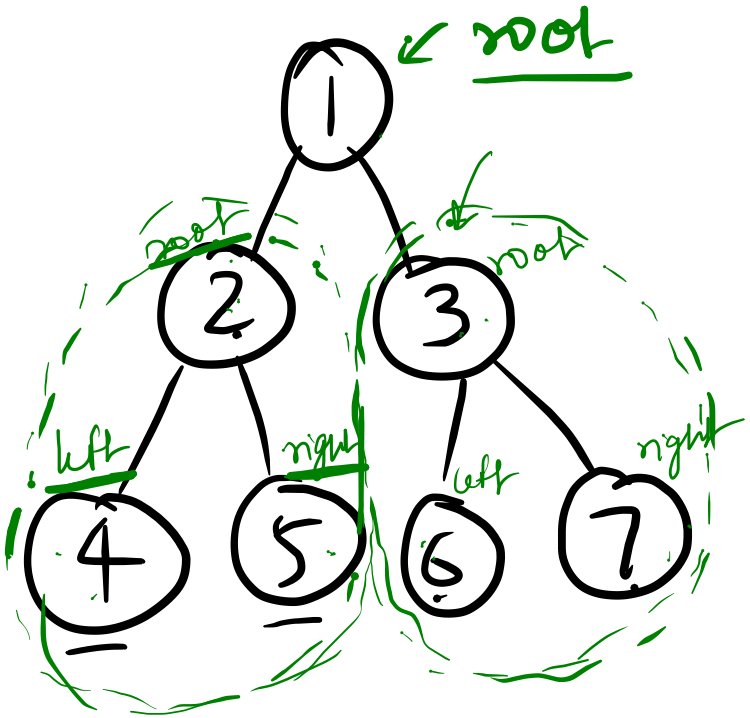
left root right

4 2 5 1 6 3 7



Preorder

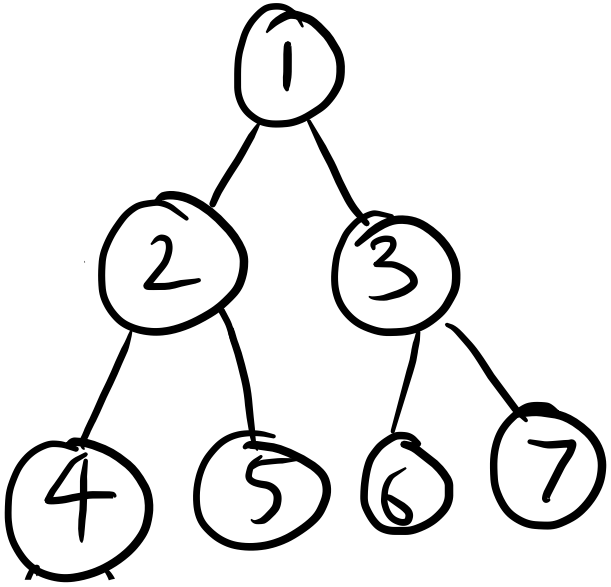
root	left	right
1	2 4 5	3 6 7



Postorder

left	right	root
4 5 2	6 7 3	1
✓	✓	✓

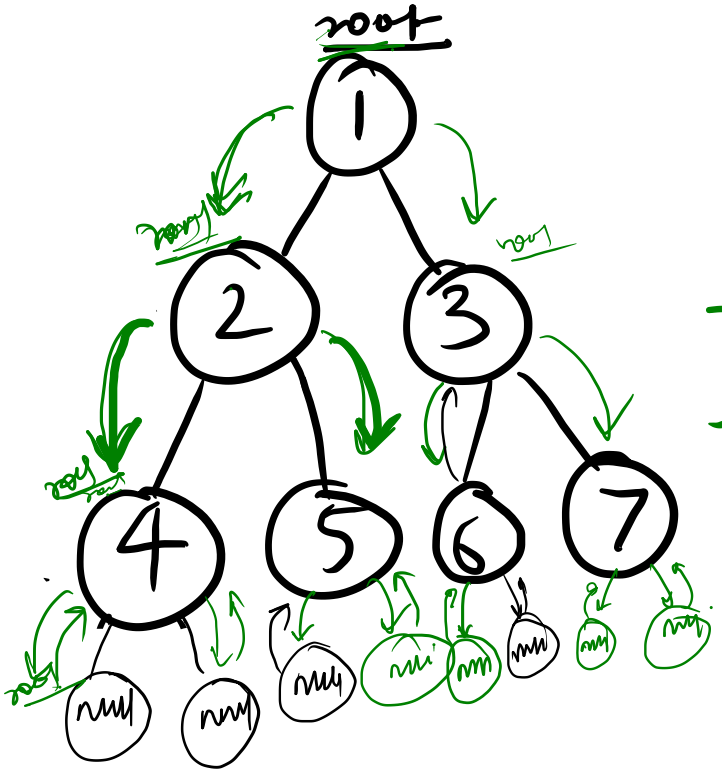
Inorder	L	root	R
preorder	root	L	R
postorder	L	R	root



① Inorder Traversal

① visit the left node
② visit the root node
③ visit the right node

{ 4 2 5 , 1 , 6 3 7 }



① Inorder Traversal

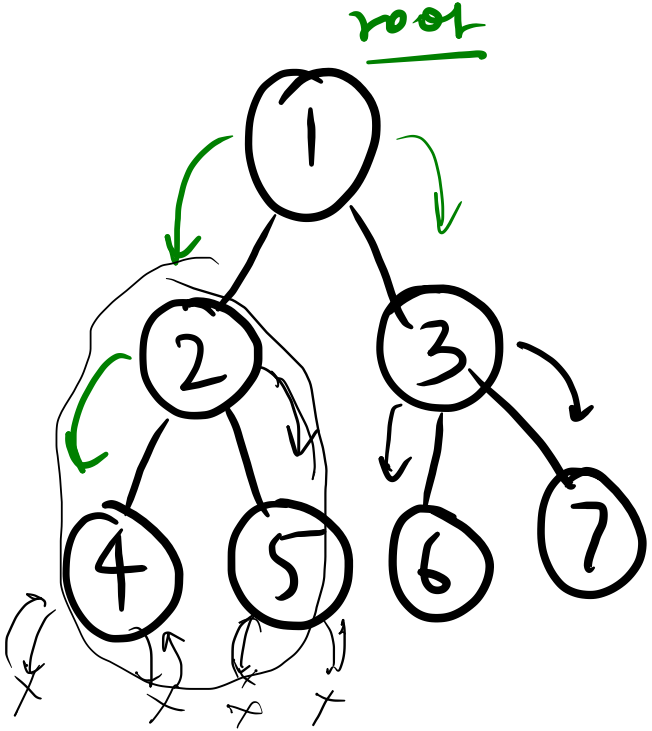
→ ① visit the left node

→ ② Add the root node

→ ③ visit the right node

⇒ { 4, 2, 5, 1, 6, 3, 7 }

result

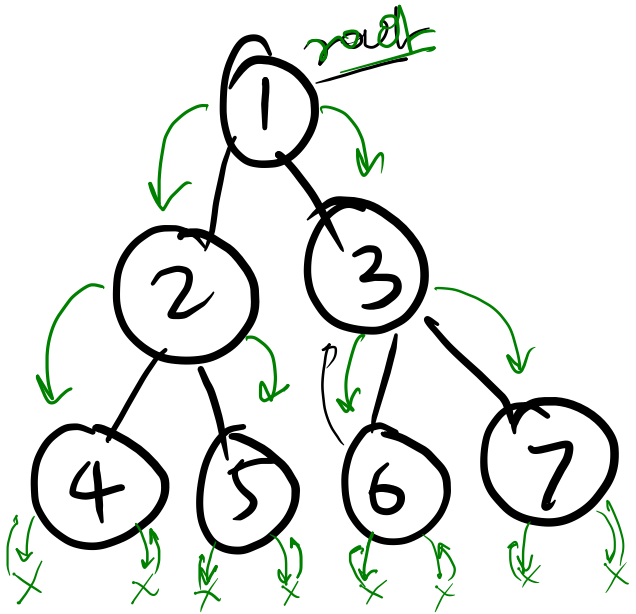


Preorder

1. Add the root
2. visit the left
3. visit the right

{ 1, 2, 4, 5, 3, 6, 7 }

result



Postorder

1. Visit the left
2. Visit the right
3. Add the root

{4, 5, 2, 6, 7, 3, 1}

result

Time Complexity = O(n)

Space Complexity = O(h)

h → height

(recursive stack space)