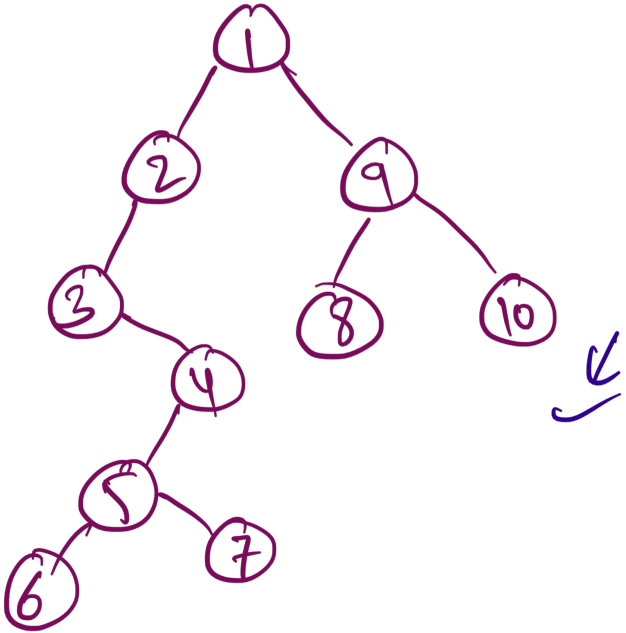
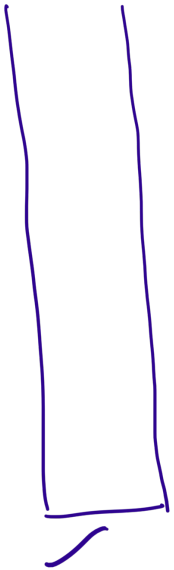
Binary Tree - II

Inorder Traversal without using Recursion





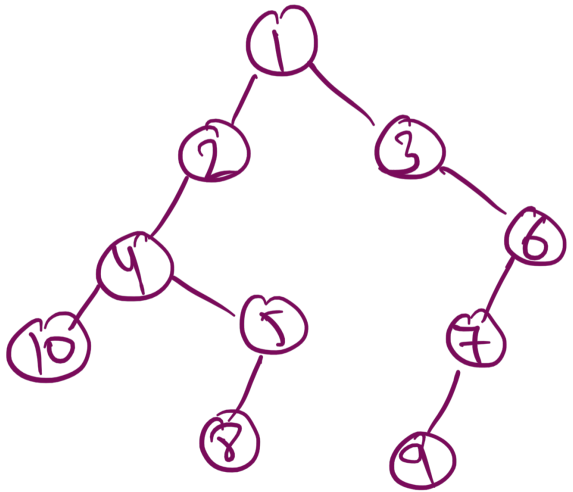
3 6 5 7 4 2 1 8 9 10

!(stack.isEmpty && curr == null)

!(a && b)

= !a || !b.

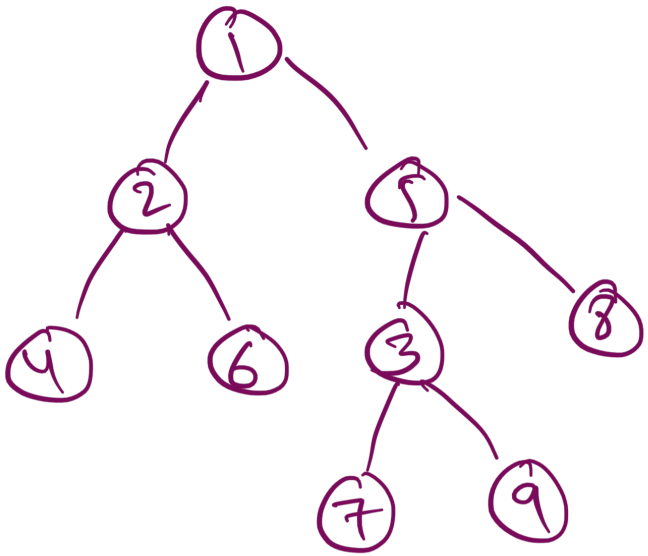
Preorder Traversal without using Recursion

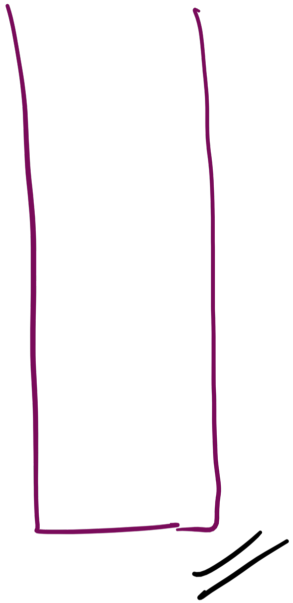


NLR

1 2 4 5 8 3 6 7 9

1 2 10 5 8 3 6 7 9



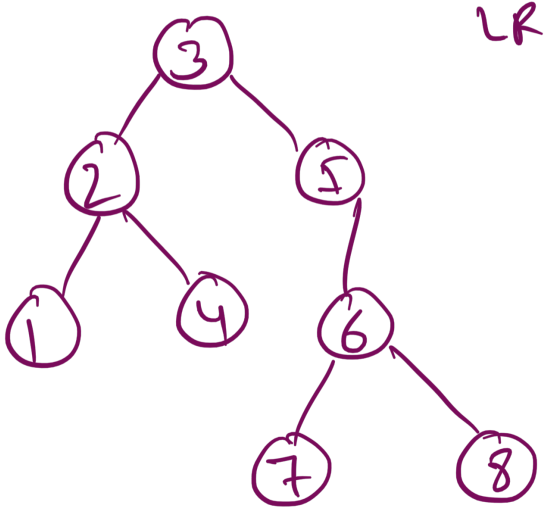


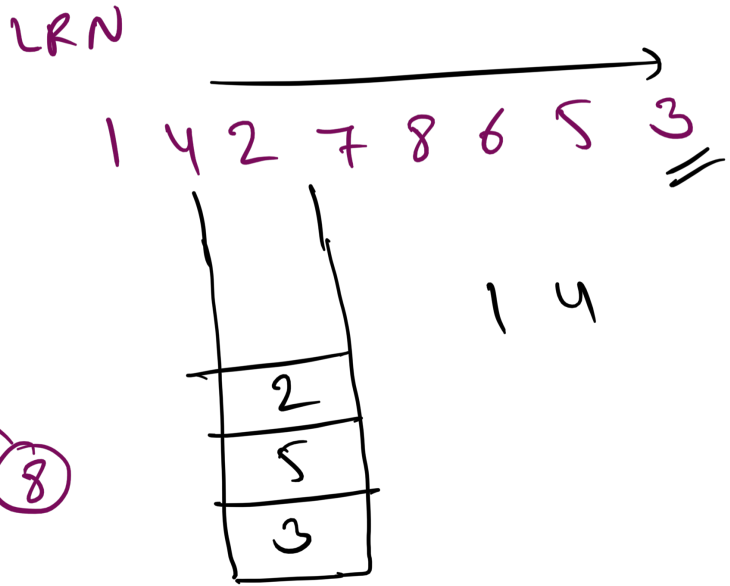
1 2 4 6 5 3 7 9 8

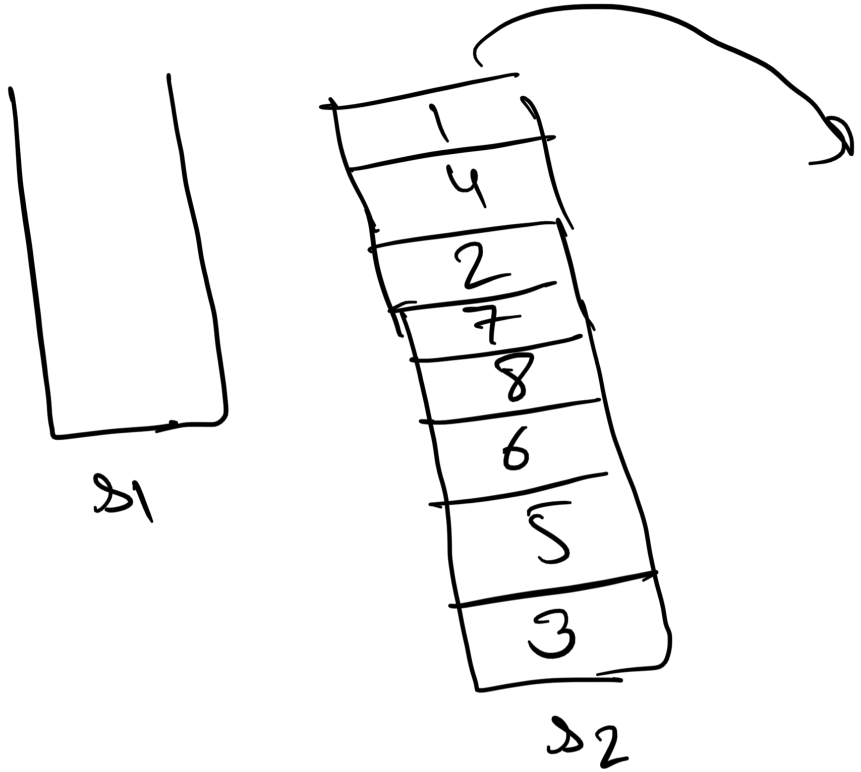
1 2 4 6 5 3 7 9 8

NLR

Postorder Traversal without using Recursion

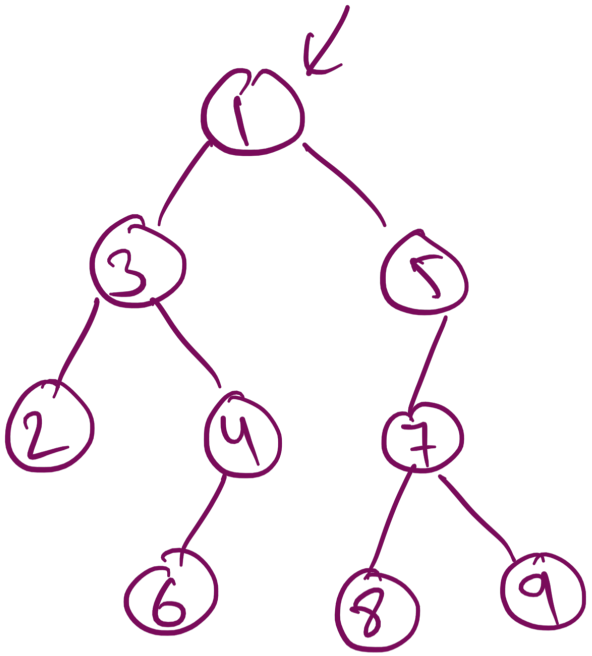


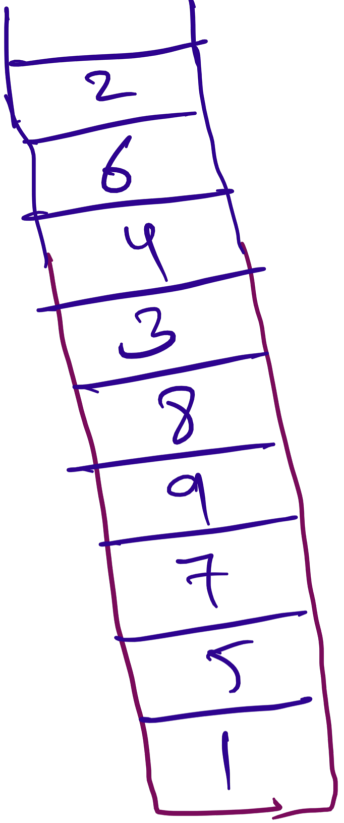





```
while (s1.isNotEmpty()){
    s2.push(s1.pop());
    s1.push(pop.left);
    s1.push(pop.right);
}
```

print s2



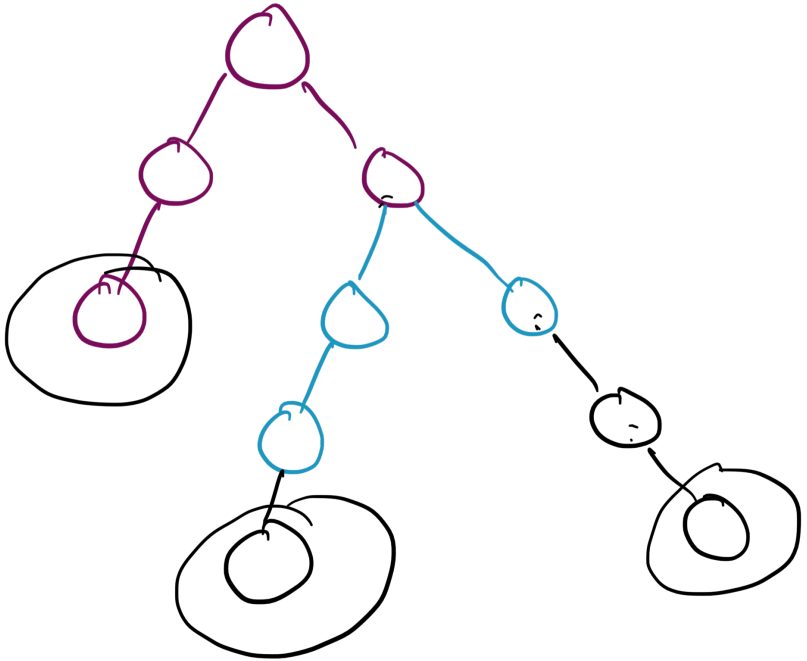


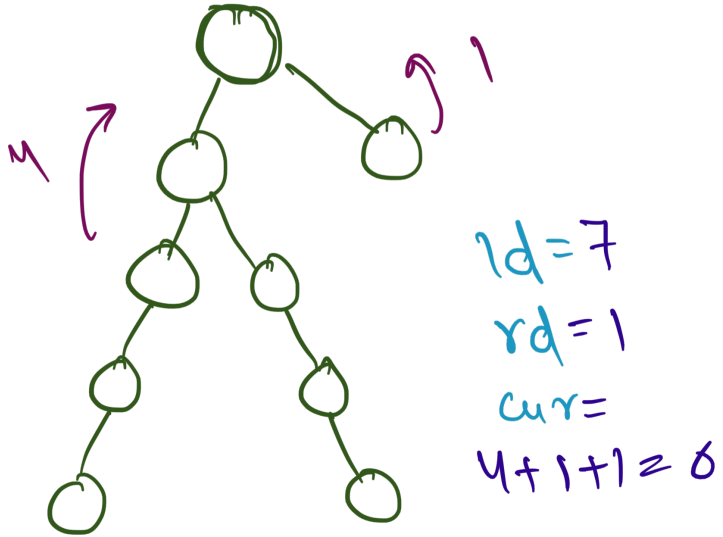
LRN

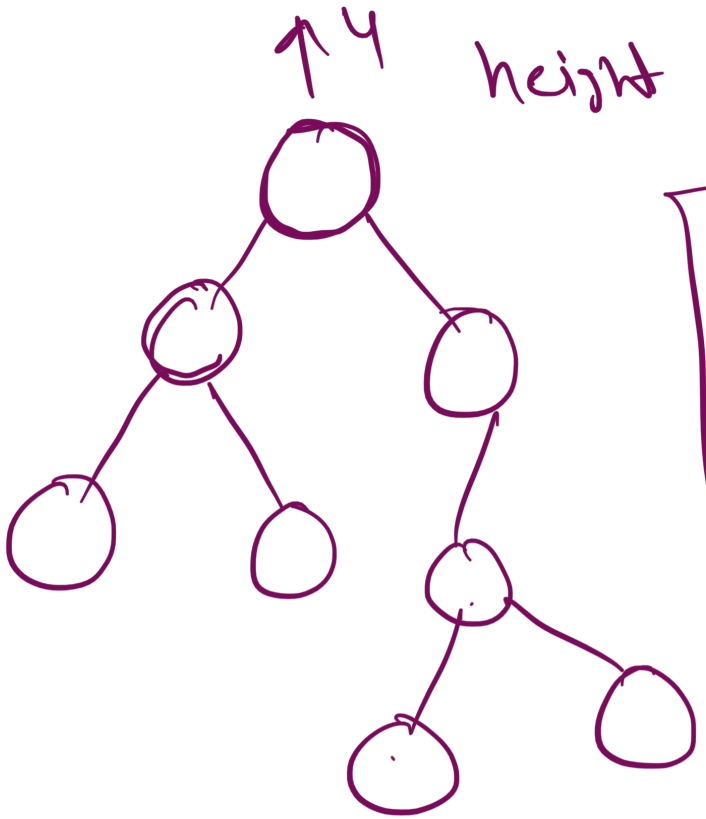
2 6 4 3 8 9 7 5 1

Diameter of a Binary Tree

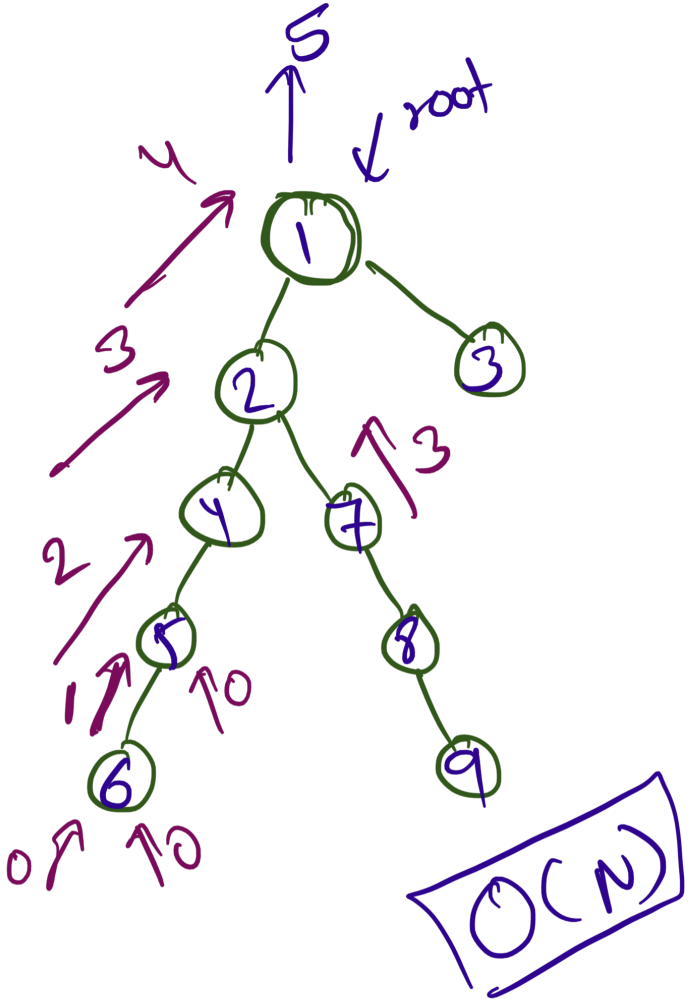
The largest distance between any two nodes in a binary tree is called diameter.

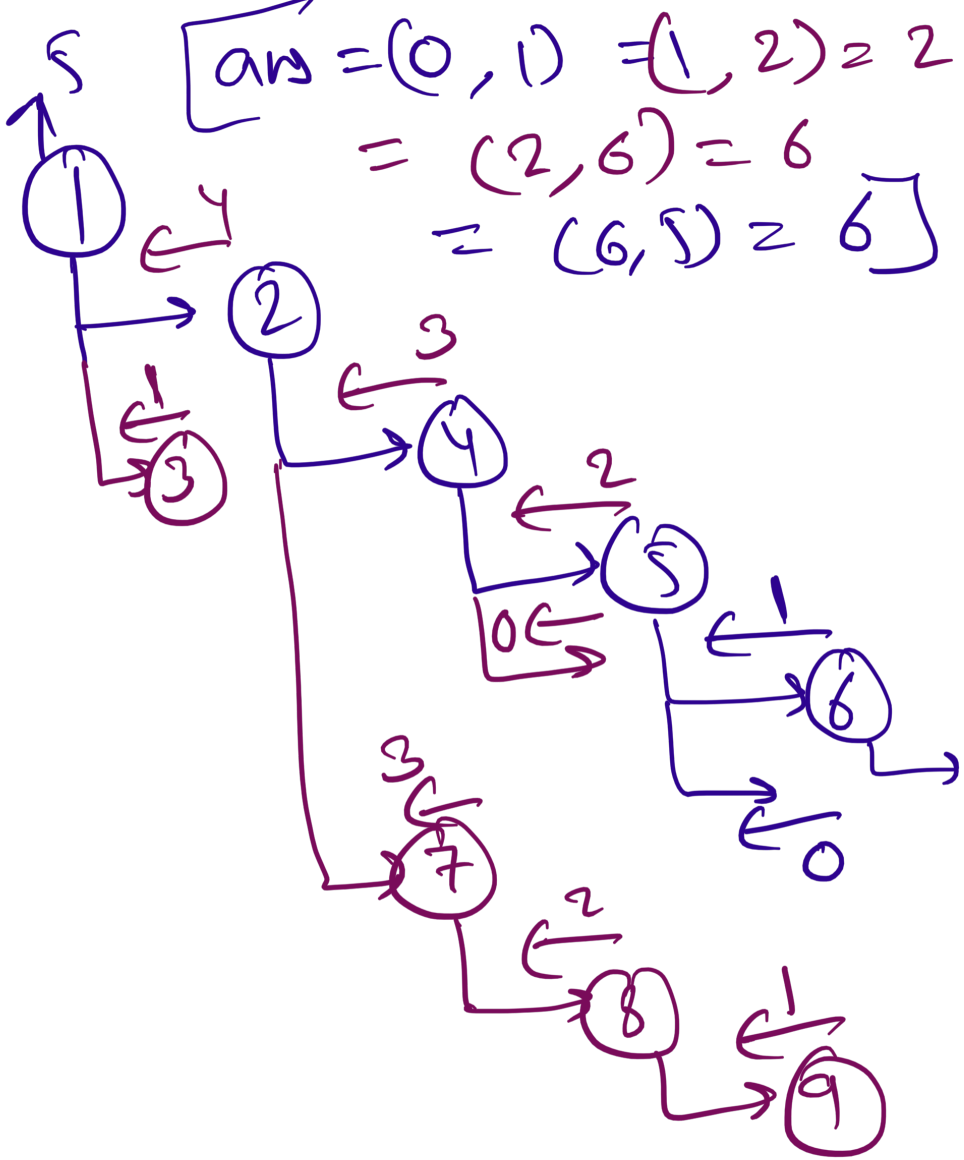




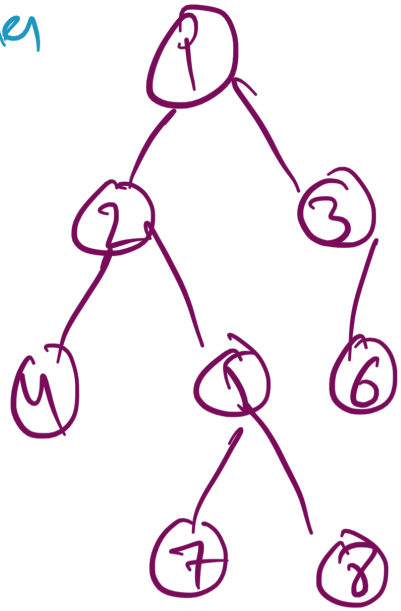

4
height
ans = 0
height (root) {
dh = height (left)
rh = height (right)
ans = Max(ans, dh+rh)
return Math.max(dh, rh)+1;
y

$O(N^2)$.


5
root
4
3
2
4
7
3
2
1
10
6
8
9
0
10
O(N)

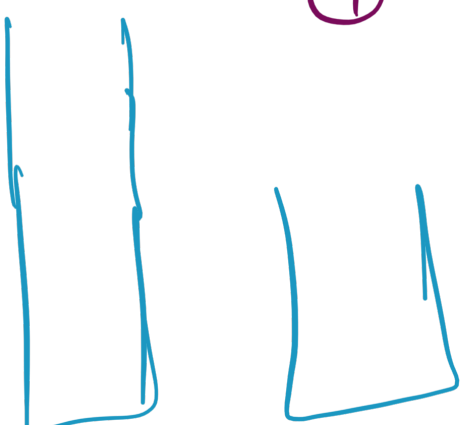

ans = (0, 1) = (1, 2) = 2
= (2, 6) = 6
= (6, 5) = 6
5
1
4
2
3
1
3
4
2
5
0
1
6
0
3
7
2
8
1
9

Spiral order Traversal:

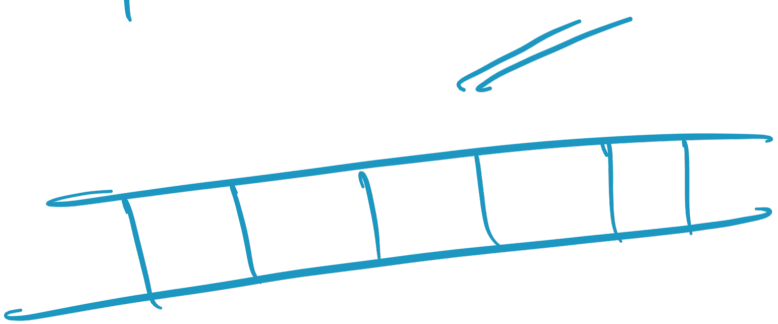


1 3 2 4 5 6 8 7

Zig-Zag.



1 3 2 4 6 8 7



Practice Problems

1. Postorder of a Binary Tree without using 2 stacks
2. Morris Traversal
3. https://www.interviewbit.com/courses/programming/tree-data-structure