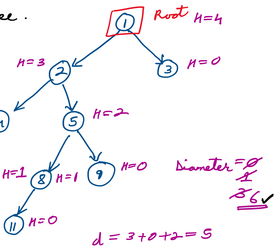
Problems on Trees

Wednesday, 29 September 2021 9:10 PM

Diameter of Tree → Longest distance (#edges) b/w any two nodes in the tree.

0→ Find diameter of binary tree.



diameter = 0;

int height(root) {

if(root == null)

return -1;

L = height(root.left);

R = height(root.right);

diameter = max(diameter, L+R+2);

return max(L,R) + 1;

}

TC = O(N)

SC = O(H)

Morris Inorder Traversal 0%0

SC = O(1)

[3, 5, 8, 9, 10, 20, 50]

Left Node Right

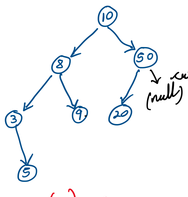
Goal → Find a way to come back to current node once we go left.

We want to come back to current node from inorder predecessor of current node.

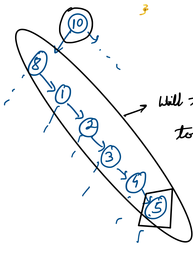
```
node predecessor(cur){
    temp = cur.left;
    while(temp.right != null &&
          temp.right != cur)
    { temp = temp.right; }
    return temp;
}
```

```
cur = root;
while(cur != null){
    if(cur.left == null){
        print(cur.data);
        cur = cur.right;
    } else {
        pre = predecessor(cur);   // cur->left->right->right-...
        if(pre.right == null){
            pre.right = cur;      // ΔO
            cur = cur.left;
        } else {
            pre.right = null;     // UNDO
            print(cur.data);
            cur = cur.right;
        }
    }
}
```

Inorder → [3 5 8 9 10 20 50]



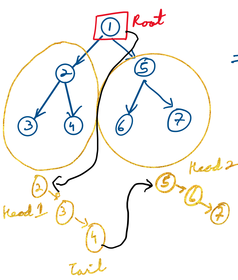
TC = O(N) ✓

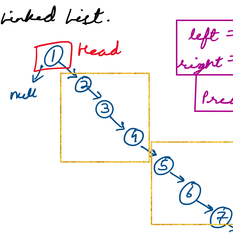


Will these nodes be travelled ever to find predecessor for any other node? No

2 times for 10.

0→ Flatten Binary Tree to linked list.





left = null

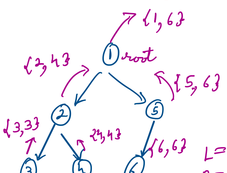
right = next ptr of LL

Preorder Traversal

TC = O(N)

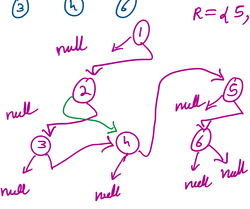
first = Head

second = tail

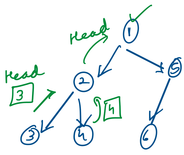


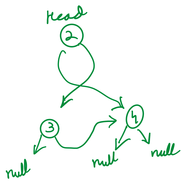
L = {2,4}

R = {5,6}




```
pair<node> flatten(root){
    if(root == null) return {null, null};
    L = flatten(root.left);
    R = flatten(root.right);
    root.left = null;
    if(L.head != null){
        root.right = L.head;
        L.tail.right = R.head;
    } else {
        root.right = R.head;
    }
    return {root,
            (R.tail != null ? R.tail :
             (L.tail != null ? L.tail :
              root))};
}
```





L = 3

R = 4