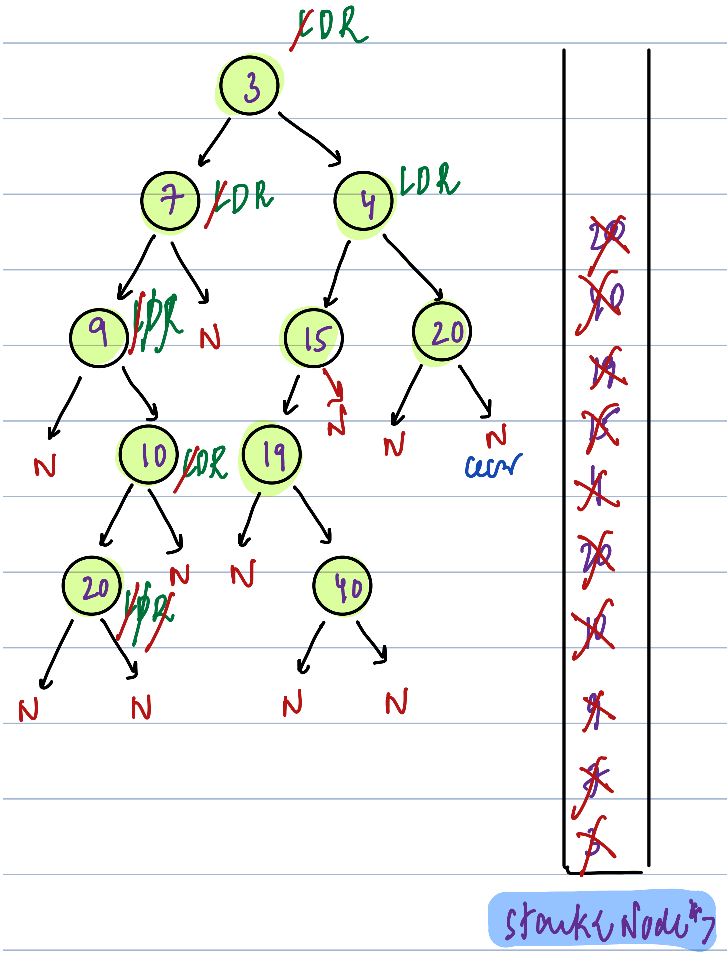
Todays Content

a. Iterative InOrder

b. Iterative Pre Order

c. Iterative PostOrder

Iterative InOrder (LDR)



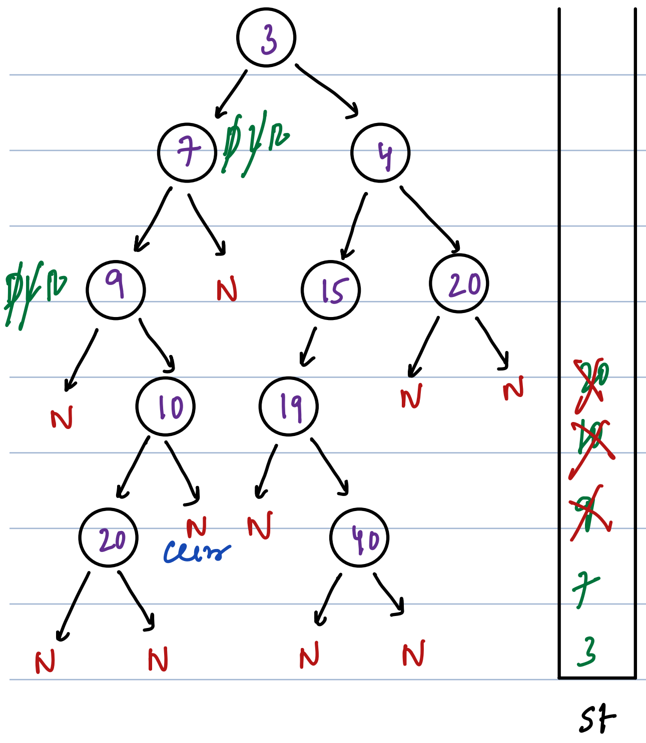
```
void inOrder(Node root){
    Node curr = root;
    stack<Node*> st;
    while( curr != null || st.size() >0 ){
        if(curr != nullptr){
            st.push(curr);
            curr = curr -> left;
        }
        else{
            curr = st.top();   ↗ if st.size==0
                                   break
            st.pop();
            print(curr -> data);
            curr=curr -> right
        }
    }
}
```

Output

9 20 10 7 3 19 40 15 4 20

Iterative PreOrder(DLR)

~~D~~ ~~L~~ R

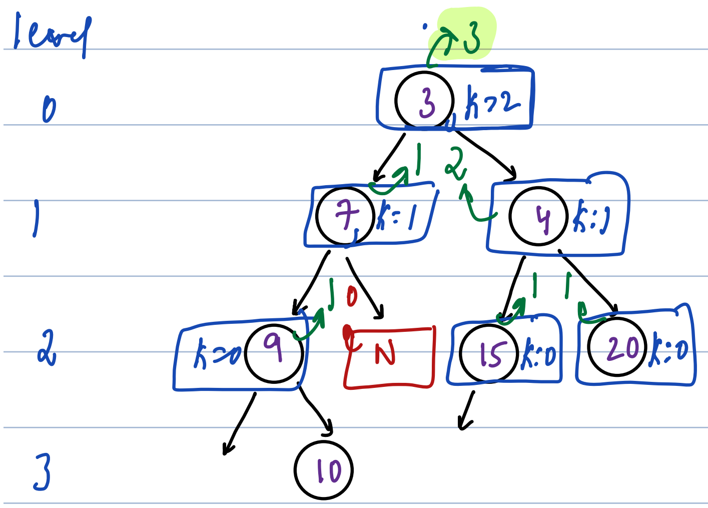


```
void preOrder(Node root){
    Node curr= root;
    stack<Node*> st;
    while(curr != null || st.size() >0 ){
        if(curr != nullptr){
            print(curr->data);
            st.push(curr);
        }   curr = curr->left;
        else{
            curr = st.top();   -> if st.size()==0
                                  break
            st.pop();
            curr=curr->right
        }
    }
}
```

3 7 9

PostOrder: TODO

Return count of nodes at level: k



```
int countk(Node root, int k){
    if(root==null){return 0}
    if( k==0){return 1}
    int l = countk(root.left, k-1)
    int r = countk(root.right, k-1)
    return l+r;
}
```