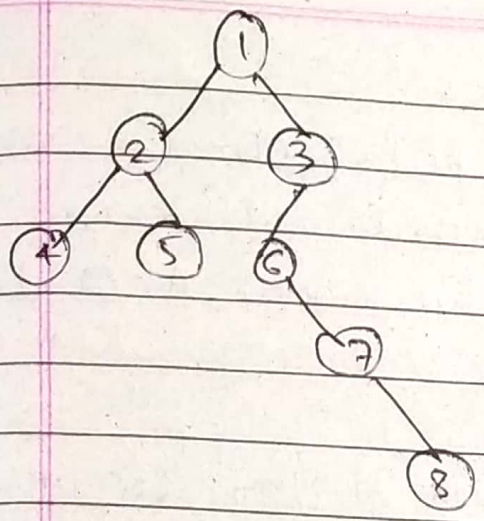
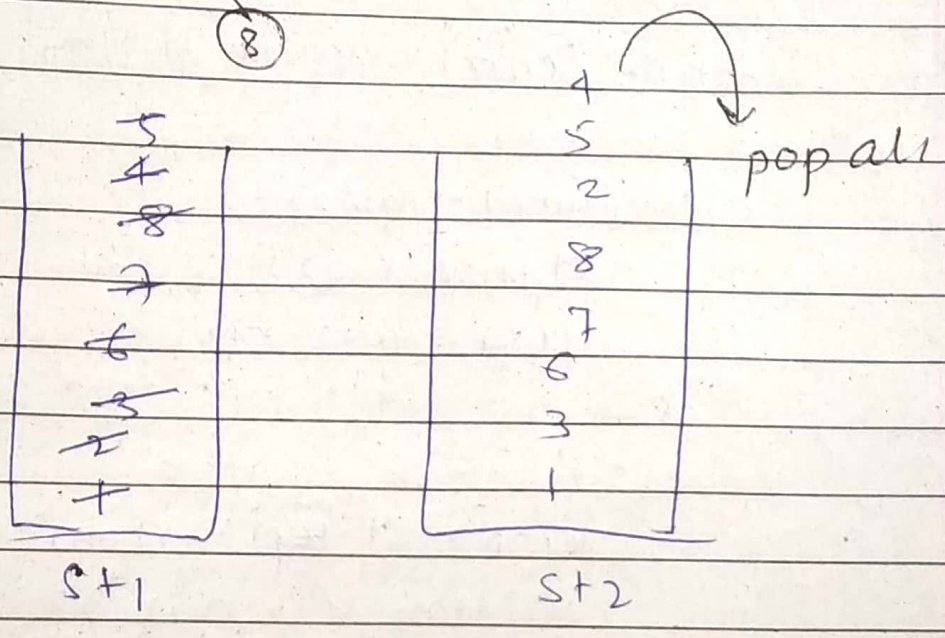
Postorder using Iteration (L R N)



postorder:

4 5 2 8 7 6 3 1



Code

```
class solution{
    public List<Node<Integer>> postOrder(TreeNode root)
{   stack<TreeNode> st1 = new Stack<>();
    Stack<TreeNode> st2 = new Stack<>();
    ArrayList<Integer> al = new ArrayList<>();
     st1.push(root);
    while(!st1.isEmpty()){
          root = st1.pop();
          st2.push(root);
          if(root.left!=null) st2.add(root.left);
          if(root.right!=null) st2.push(root.right);
    }
    while(!st2.isEmpty()){
          al.add(st1.pop().val);
    }
}
```

(using 1 stack)

```
class solution{
public List<Integer> postOrder (TreeNode root){
Stack <TreeNode> st = new Stack<>();
ArrayList <Integer> al = new ArrayList<>
Node curr = root;

while (curr!==null || !st.isEmpty())
{
  if (curr!=null){
    st.push(curr);
    curr=curr.left;
  }
  else
    temp = st.peek().right;
    if (temp == null){
    temp = st.pop();
    st.pop();
    al.add(temp.val);
    while(!st.isEmpty() && temp.
                temp=st.peek().right
    {
      temp = st.peek();
      st.pop();
      al.add(temp.val);
    }
  }
  else
    curr = temp
```