public static class Node {

int data;

Node left;

Node right;

Node(int data, Node left, Node right) {

this.data = data;

this.left = left;

this.right = right;

}

}

public static class Pair {

Node node;

int state;

Pair(Node node, int state) {

this.node = node;

this.state = state;

}

}

public static Node construct(Integer[] arr)

{

Node root = new Node(arr[0], null, null);

Pair rtp = new Pair(root, 1);

Stack<Pair> st = new Stack<>();

st.push(rtp);

int idx = 0;

while (st.size() > 0) {

Pair top = st.peek();

if (top.state == 1) {

idx++;

if (arr[idx] != null) {

top.node.left = new Node(arr[idx], null, null);

Pair lp = new Pair(top.node.left, 1);

st.push(lp);

} else {

top.node.left = null;

}

top.state++;

} else if (top.state == 2) {

idx++;

if (arr[idx] != null) {

top.node.right = new Node(arr[idx], null, null);

Pair rp = new Pair(top.node.right, 1);

st.push(rp);

} else {

top.node.right = null;

}

top.state++;

} else {

st.pop();

}

}

return root;

}

public static void display(Node node)

{ if (node == null) {

return; }

String str = "";

str += node.left == null ? "." : node.left.data + "";

str += " <- " + node.data + " -> ";

str += node.right == null ? "." : node.right.data + "";

System.out.println(str);

display(node.left);

display(node.right);

}

public static int size(Node node)

{ if(node==null)

{ return 0; }

int n1=size(node.left);

int n2=size(node.right);

int ans=n1+n2+1;

return ans; }

public static int sum(Node node)

{

if(node==null)

{ return 0; }

int n1=sum(node.left);

int n2=sum(node.right);

int ans=n1+n2+node.data;

return ans;

}

public static int max(Node node)

{ if(node==null)

{ return 0; }

int n1=max(node.left);

int n2=max(node.right);

return Math.max(node.data,Math.max(n1,n2));

}

public static int height(Node node)

{

if(node==null)

{ return -1; }

int n1=height(node.left);

int n2=height(node.right);

int ans=Math.max(n1,n2)+1;

return ans;

}

public static voidTravesel(Node root)

{

if(root==null)

{

return;

}

System.out.print("Preorder "+root.data+" ");//Preorder

inorder(root.left);

System.out.print("Inorder "+root.data+" ");//in

inorder(root.right);

System.out.print("Postorder "+root.data+" ");//post

}