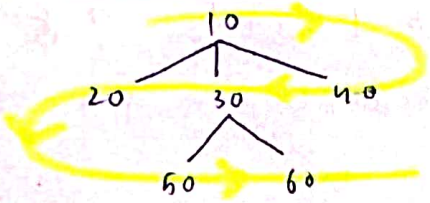
⑧ LEVELORDER LINEWISE ZIG-ZAG TRAVERSAL IN GENERIC TREE

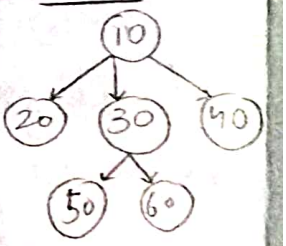


Output:

10
40 30 20
50 60

- Hum issme Stack Data Structure Use krengy!
- Hum 2 stack banayeng < main stack, child stack, aur ek flag use krengy!
- Issme bhi RPA use krengy! Remove → Print → Add (Remove, Print: main stack; Add: child stack)
- Phele main stack me root daaldengy.
- Odd levels par hume left to right jana hai! Agar hum children ko bhi left to right order me add krey child stack me toh stack se vo right to left bahar nikalengy! Same goes with even levels!
- Even Levels par hum right to left jana hai! Agar hum children ko bhi right to left order me add krey child stack me toh stack se vo left to right nikalengy!

START



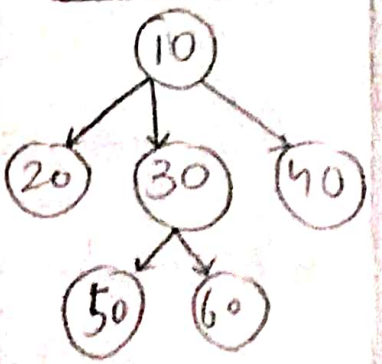
OUTPUT

10
40 30 20
50 60

Code

```
public static void levelOrderLinewiseZZ(Node node){
 Stack <Node> ms = new Stack<>();
 Stack <Node> cs = new Stack<>();
 ms.push(node);
 int level = 1;
 while(ms.size() > 0){
    Node temp = ms.pop();
    System.out.print(temp.data+" ");
    if(level %. 2 == 1){
       for(int i=0; i< temp.children.size(); i++){
        cs.push(temp.children.get(i));
       }
    }else{
        for(int i=temp.children.size()-1; i>=0; i--){
         cs.push(temp.children.get(i));
        }
    }
    if(ms.size() == 0){
      ms= cs;
      cs = new stack<>();
      level ++;
      System.out.println();
    }
  }
}
```

START

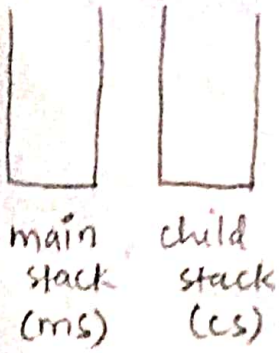


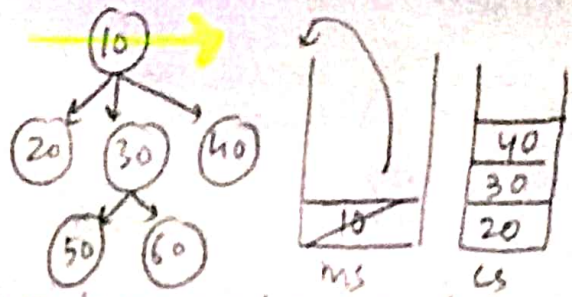
OUTPUT

10

40 30 20

50 60





∴ 10 is removed and printed

∴ Level 1 (left → right)

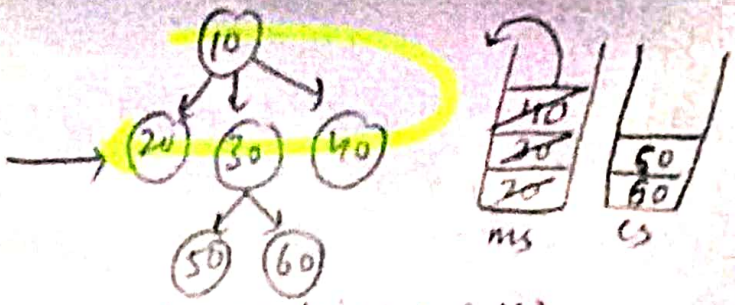
∴ childrens of nodes of level 1 will be add to (cs) in left → right order

∴ Jab (ms) empty hogi tab

ms = cs

cs = new

∴ move to next line



∴ Now Level 2 (right to left)

∴ childrens of nodes of Level 2 will be added to (cs) in right to left order.

∴ 40 removed + printed (No child)

∴ 30 removed + printed (child's are added to (cs) → 50, 60

∴ 20 removed + printed (No child)

∴ (ms) = empty ← ms = cs, cs = new, Enter next line

END

∴ Now level 3 (left to right)

∴ childrens of nodes of Level 3 will be added to (cs) in left to right order

∴ 50 removed + printed (No child)

∴ 60 removed + printed (No child)