**Bottom View of Binary Tree**

Given a binary tree, print the bottom view from left to right.  
A node is included in bottom view if it can be seen when we look at the tree from bottom.

                      20  
                    /    \  
                  8       22  
                /   \        \  
              5      3       25  
                    /   \        
                  10    14

For the above tree, the bottom view is 5 10 3 14 25.  
If there are **multiple**bottom-most nodes for a horizontal distance from root, then print the later one in level traversal. For example, in the below diagram, 3 and 4 are both the bottommost nodes at horizontal distance 0, we need to print 4.

                      20  
                    /    \  
                  8       22  
                /   \     /   \  
              5      3 4     25  
                     /    \        
                 10       14

For the above tree the output should be 5 10 4 14 25.

Solution:-

//User function Template for Java

class Solution

{

//Function to return a list containing the bottom view of the given tree.

void traverse(TreeMap<Integer,ArrayList<Integer>> tm,Node node,int vlevel,int hlevel){

if(node==null)

return;

else{

if(!tm.containsKey(vlevel)){

tm.put(vlevel,new ArrayList<>());

tm.get(vlevel).add(hlevel);

tm.get(vlevel).add(node.data);

}

else if(hlevel>=tm.get(vlevel).get(0)){

tm.get(vlevel).set(0,hlevel);

tm.get(vlevel).set(1,node.data);

}

traverse(tm,node.left,vlevel-1,hlevel+1);

traverse(tm,node.right,vlevel+1,hlevel+1);

}

}

public ArrayList <Integer> bottomView(Node root)

{

// Code here

ArrayList<Integer> ar=new ArrayList<>();

TreeMap<Integer,ArrayList<Integer>> tm=new TreeMap<>();

traverse(tm,root,0,0);

//System.out.println(tm);

int key=tm.firstKey();

while(tm.containsKey(key)){

ar.add(tm.get(key).get(1));

key++;

}

return ar;

}

}