**Check for Balanced Tree:-**

Given a binary tree, find if it is height balanced or not.   
A tree is height balanced if difference between heights of left and right subtrees is **not more than one** for all nodes of tree.

**A height balanced tree**  
        1  
     /     \  
   10      39  
  /  
5

**An unbalanced tree**  
        1  
     /      
   10     
  /  
5

**Example 1:**

**Input:**

      1

   /

   2

   \

    3

**Output:** 0

**Explanation:** The max difference in height

of left subtree and right subtree is 2,

which is greater than 1. Hence unbalanced

**Example 2:**

**Input:**

       10

    /   \

   20   30

  /   \

40   60

**Output:** 1

**Explanation:** The max difference in height

of left subtree and right subtree is 1.

Hence balanced.

**Your Task:**  
You don't need to take input. Just complete the function**isBalanced()**that takes root **node**as parameter and returns **true,**if the tree is balanced else returns **false**.

**Constraints:**  
1 <= Number of nodes <= 105  
0 <= Data of a node <= 106

**Expected time complexity:**O(N)  
**Expected auxiliary space:**O(h) , where h = height of tree