# PROBLEM

## K distance from root

Easy Accuracy: 52.18% Submissions: 88K+ Points: 2



Given a binary tree having **n** nodes and an integer **k**. Print all nodes that are at distance k from the root (root is considered at distance 0 from itself). Nodes should be printed from **left to right**.

**Example 1:**

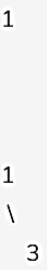
**Input:**
k = 0



**Output:**
1
**Explanation:**
1 is the only node which is 0 distance from the root 1.

**Example 2:**

**Input:**
k = 3



**Output:**
5 3
**Explanation:**
5 and 3 are the nodes which are at distance 3 from the root 3.
Here, returning 3 5 will be **incorrect**.

**Your Task:**
You don't have to take input. Complete the function **Kdistance()** that accepts **root** node and **k** as parameters and returns the value of the nodes that are at a distance k from the root.

**Expected Time Complexity:** O(n).
**Expected Auxiliary Space:** O(Height of the Tree).

**Constraints:**
$1 <= n <= 10^4$
$0 <= k <= 30$

## CODE

```
    #User function Template for python3



'''
class Node:
    def _init_(self,val):
        self.data = val
        self.left = None
        self.right = None
'''
class Solution:
    def KDistance(self,root,k):
        queue = [[root, 0]]
        res = []
        for [node, depth] in queue:
            if(depth == k):
                res.append(node.data)
            node.left and queue.append([node.left,
depth+1])
            node.right and queue.append([node.right,
depth+1])
        return res
    '''
    :param root: root of given tree.
    :param k: distance k from root
    :return: list of all nodes that are at distance k from
root.
    '''
    # code here
```