**K distance from root**

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

  1

  / \

  3 2

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

**Example 2:**

**Input:**

k = 3

  1

  /

  2

  \

  1

  / \

  5 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** nodeand **k**as parameters and returnsthe valueof the nodesthat are at a distance k from the root.

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

**Constraints:**  
1 <= n <= 104  
0 <= k <= 30

class Solution

{

public:

// function should print the nodes at k distance from root

vector<int> Kdistance(struct Node \*root, int k)

{

vector<int> v;

if(root == NULL){

return v;

}

queue<pair<Node\*, int>>q;

q.push({root,0});

while(!q.empty()){

auto it = q.front();

q.pop();

Node\* temp = it.first;

int val = it.second;

if(val == k){

v.push\_back(temp->data);

}

if(temp->left != NULL){

q.push({temp->left,val + 1});

}

if(temp->right != NULL){

q.push({temp->right,val + 1});

}

}

return v;

}

};

Link : <https://www.geeksforgeeks.org/problems/k-distance-from-root/1>