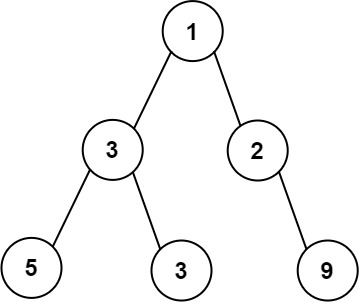
Given the root of a binary tree, return *the****maximum width****of the given tree*.

The **maximum width** of a tree is the maximum **width** among all levels.

The **width** of one level is defined as the length between the end-nodes (the leftmost and rightmost non-null nodes), where the null nodes between the end-nodes are also counted into the length calculation.

It is **guaranteed** that the answer will in the range of **32-bit** signed integer.

**Example 1:**

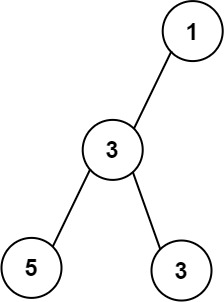


**Input:** root = [1,3,2,5,3,null,9]

**Output:** 4

**Explanation:** The maximum width existing in the third level with the length 4 (5,3,null,9).

**Example 2:**

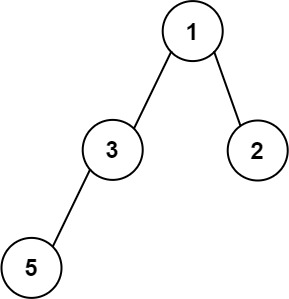


**Input:** root = [1,3,null,5,3]

**Output:** 2

**Explanation:** The maximum width existing in the third level with the length 2 (5,3).

**Example 3:**



**Input:** root = [1,3,2,5]

**Output:** 2

**Explanation:** The maximum width existing in the second level with the length 2 (3,2).

**Solution:**

/\*\*

\* Definition for a binary tree node.

\* public class TreeNode {

\* int val;

\* TreeNode left;

\* TreeNode right;

\* TreeNode() {}

\* TreeNode(int val) { this.val = val; }

\* TreeNode(int val, TreeNode left, TreeNode right) {

\* this.val = val;

\* this.left = left;

\* this.right = right;

\* }

\* }

\*/

class Solution {

class Pair{

TreeNode node;

int num;

Pair(TreeNode \_node, int \_num){

node = \_node;

num = \_num;

}

}

public int widthOfBinaryTree(TreeNode root) {

if(root == null)

return 0;

Queue<Pair> q = new LinkedList<>();

q.add(new Pair(root, 0));

int ans=0, first, last;

while(!q.isEmpty()){

int min = q.peek().num;

int size = q.size();

first = 0; last = 0;

for(int i=0;i<size;i++){

int cur\_id = q.peek().num-min;

TreeNode node = q.poll().node;

if(i==0) first = cur\_id;

if(i==size-1) last = cur\_id;

if(node.left != null)

q.add(new Pair(node.left, 2\*cur\_id+1));

if(node.right != null)

q.add(new Pair(node.right, 2\*cur\_id+2));

}

ans = Math.max(ans, last-first+1);

}

return ans;

}

}