# 111、二叉树的最小深度（Minimum Depth of Binary Tree）

## 题目：

给定一个二叉树，找出其最小深度。

最小深度是从根节点到最近叶子节点的最短路径上的节点数量。

**说明:** 叶子节点是指没有子节点的节点。

**示例:**

给定二叉树 [3,9,20,null,null,15,7],

3   
 / \   
 9 20   
 / \   
 15 7

返回它的最小深度 2.

## 解答：

法一：

|  |
| --- |
| public int MinDepth(TreeNode root)  {  if (root == null)  {  return 0;  }  if (root.left == null || root.right == null)  {  return Math.Max(MinDepth(root.left), MinDepth(root.right)) + 1;  }  return Math.Min(MinDepth(root.left), MinDepth(root.right)) + 1;  }  /\* 最小的深度  \* 1  \* / \ 4 > 0 > 1  \* 1 1 0  \* / \ 3 > 1 > 0(这里就会出现问题所以只有一边的需要+1)  \* 1 null 0  \* / \ 改：3 > 1 > 2  \* null null 0  \* 2 > 2 > 1  \* 1  \*  \*条件：1、min(left,right) + 1  \* 2、只有一个（必须）min(left,right) + 1  \* 3、null 0  \*/ |

法一：

|  |
| --- |
| public int MinDepth1(TreeNode root)  {  if (root == null)  {  return 0;  }  int count = 0;  Queue<TreeNode> rootQueue = new Queue<TreeNode>();  rootQueue.Enqueue(root);  while (rootQueue.Count > 0)  {  count++;  int lenrootQueue = rootQueue.Count;  for (int i = 0; i < lenrootQueue; i++)  {  TreeNode curRoot = rootQueue.Dequeue();  if (curRoot.left == null && curRoot.right == null)  {  return count;  }  if (curRoot.left != null)  {  rootQueue.Enqueue(curRoot.left);  }  if (curRoot.right != null)  {  rootQueue.Enqueue(curRoot.right);  }  }  }  return 0;  }  /\* 最小的深度 count rootQueue curRoot  \* 3 1) 1 3 3  \* / \ 2) 2 9 20 9 > 返回2  \* 9 20  \* / \  \* 15 7  \*/ |