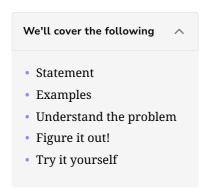
C

Maximum Depth of Binary Tree

Try to solve the Maximum Depth of Binary Tree problem.



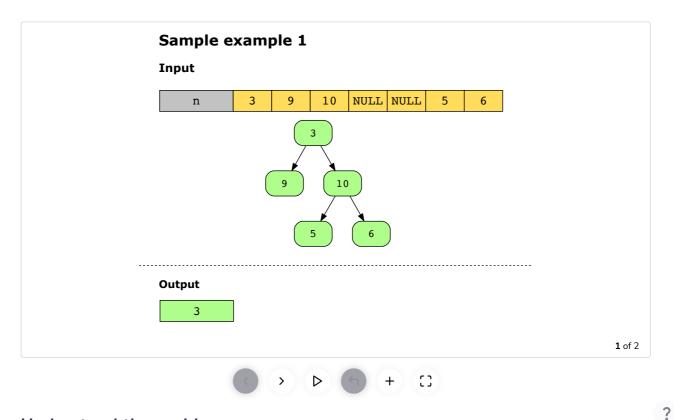
Statement

Given the root of a binary tree, return its maximum depth. A binary tree's maximum depth is the number of nodes along the longest path from the root node down to the farthest leaf node.

Constraints:

- The number of nodes in the tree is in the range $[0, 10^4]$.
- $-100 \le \text{node.data} \le 100$

Examples



Understand the problem

Let's take a moment to make sure you've correctly understood the problem. The quiz below helps you check if you're solving the correct problem:

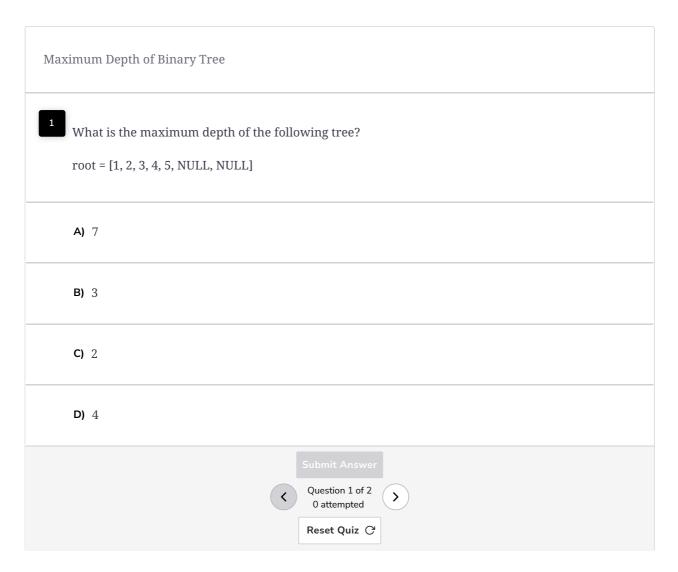
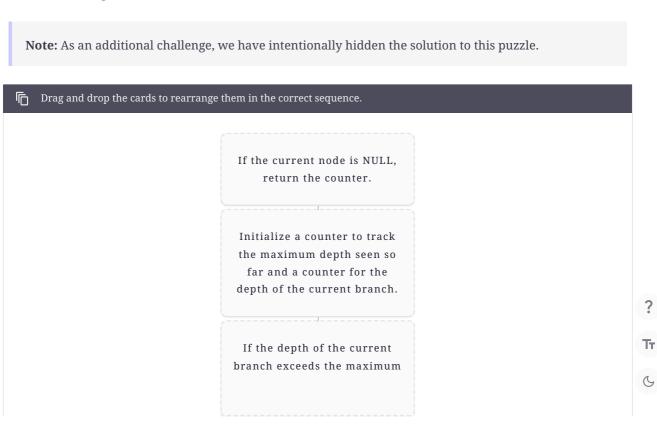


Figure it out!

We have a game for you to play. Rearrange the logical building blocks to develop a clearer understanding of how to solve this problem.



depth seen so far, update the maximum depth. Else, if the current node exists, check the depth of the left subtree and the depth of the right subtree. Compare the depths of the left and right subtrees, and select the greater of the two. Add 1 and update this as the depth of this branch. When all branches have been explored, return the maximum depth seen so far. Submit Reset

Try it yourself

Implement your solution in main. java in the following coding playground.

Note: We have left the solution to this challenge as an exercise for you. You may try to translate the logic of the solved puzzle into a coded solution.

```
Java

usercode > main.java

1 // Definition of a binary tree node class
```

■ >

```
5 //
          TreeNode<T> right;
 6
 7 //
          TreeNode(T data) {
 8 //
              this.data = data;
9 //
              this.left = null;
10 //
              this.right = null;
11 //
12 // }
13
14 import java.util.*;
15 import ds_v1.BinaryTree.TreeNode;
16
17 public class Main{
18
       public static int maxDepth(TreeNode<Integer> root) {
19
        // Your code will replace the placeholder return statement.
20
21
        return 0;
22
     }
23 }
```

?

Tτ

6

