Problem 1339: Maximum Product of Splitted Binary Tree

Problem Information

Difficulty: Medium
Acceptance Rate: 48.08%
Paid Only: No
Tags: Tree, Depth-First Search, Binary Tree

Problem Description

Given the `root` of a binary tree, split the binary tree into two subtrees by removing one edge such that the product of the sums of the subtrees is maximized.

Return _the maximum product of the sums of the two subtrees_. Since the answer may be too large, return it **modulo** `109 + 7`.

Note that you need to maximize the answer before taking the mod and not after taking it.

Example 1:

![](https://assets.leetcode.com/uploads/2020/01/21/sample_1_1699.png)

Input: root = [1,2,3,4,5,6] **Output:** 110 **Explanation:** Remove the red edge and get 2 binary trees with sum 11 and 10. Their product is 110 (11*10)

Example 2:

![](https://assets.leetcode.com/uploads/2020/01/21/sample_2_1699.png)

Input: root = [1,null,2,3,4,null,null,5,6] **Output:** 90 **Explanation:** Remove the red edge and get 2 binary trees with sum 15 and 6.Their product is 90 (15*6)

Constraints:

* The number of nodes in the tree is in the range `[2, 5 * 104]`. * `1 <= Node.val <= 104`

Code Snippets

C++:

```
/**
* Definition for a binary tree node.
* struct TreeNode {
* int val;
* TreeNode *left;
* TreeNode *right;
* TreeNode() : val(0), left(nullptr), right(nullptr) {}
* TreeNode(int x) : val(x), left(nullptr), right(nullptr) {}
* TreeNode(int x, TreeNode *left, TreeNode *right) : val(x), left(left),
right(right) {}
* };
*/
class Solution {
public:
int maxProduct(TreeNode* root) {

}
};
```

Java:

```
/**
* 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 {
public int maxProduct(TreeNode root) {

}
}
```

Python3:

```
# Definition for a binary tree node.
# class TreeNode:
# def __init__(self, val=0, left=None, right=None):
# self.val = val
# self.left = left
# self.right = right
class Solution:
def maxProduct(self, root: Optional[TreeNode]) -> int:
```