1339. Maximum Product of Splitted Binary Tree

Solved

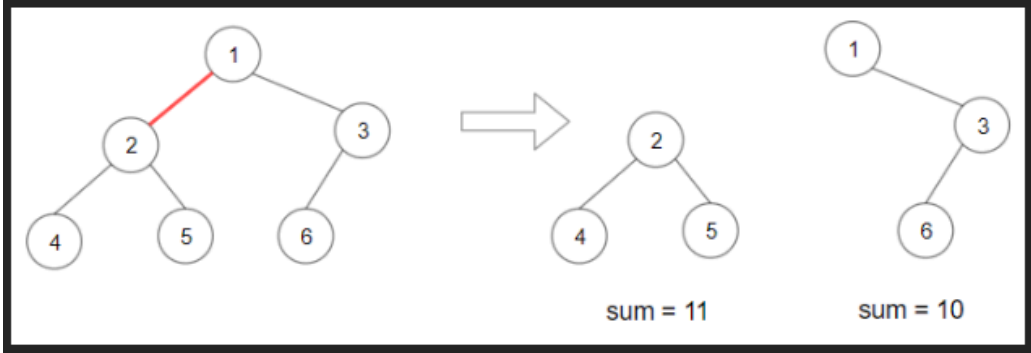
Medium | Topics | Companies | Hint

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** $10^9 + 7$.

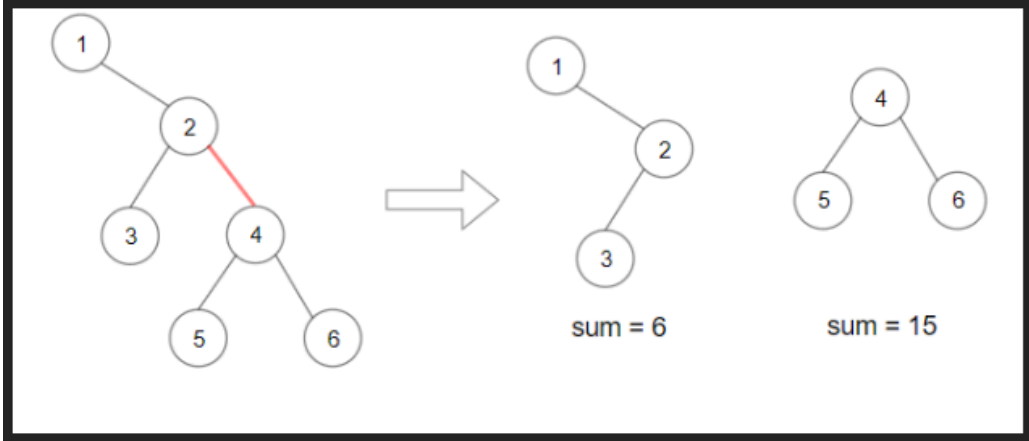
Note that you need to maximize the answer before taking the mod and not after taking it.

Example 1:




```
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:




```
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 $[2, 5 * 10^4]$.
- $1 <= Node.val <= 10^4$

Python:

```
# 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:
        ans, total=-inf, 0
        def dfs(root):
            nonlocal ans, total
            if not root: return 0
            Sum=root.val+dfs(root.left)+dfs(root.right)
            ans=max(ans, (total-Sum)*Sum)
            return Sum
        total=dfs(root)
        dfs(root)
        return ans%(10**9+7)
```

JavaScript:

```
var maxProduct = function(root) {
    const MOD = 1000000007;
    let total = 0, best = 0;

    const sum = (node) => {
        if (!node) return 0;
        return node.val + sum(node.left) + sum(node.right);
    };

    total = sum(root);

    const dfs = (node) => {
        if (!node) return 0;
        const s = node.val + dfs(node.left) + dfs(node.right);
        best = Math.max(best, s * (total - s));
        return s;
    };

    dfs(root);
    return best % MOD;
};
```

Java:

```
class Solution {
    long total = 0, best = 0;
    static final int MOD = 1_000_000_007;

    long sum(TreeNode node) {
        if (node == null) return 0;
        return node.val + sum(node.left) + sum(node.right);
    }

    long dfs(TreeNode node) {
        if (node == null) return 0;
        long s = node.val + dfs(node.left) + dfs(node.right);
        best = Math.max(best, s * (total - s));
        return s;
    }

    public int maxProduct(TreeNode root) {
        total = sum(root);
        dfs(root);
        return (int)(best % MOD);
```

```
    }
}
```