Problem 1666: Change the Root of a Binary Tree

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

Given the

root

of a binary tree and a

leaf

node, reroot the tree so that the

leaf

is the new root.

You can reroot the tree with the following steps for each node

cur

on the path

starting from the

leaf

up to the

root

excluding the root

:

If

cur

has a left child, then that child becomes

cur

's right child.

cur

's original parent becomes

cur

's left child. Note that in this process the original parent's pointer to

cur

becomes

null

, making it have at most one child.

Return

the new root

of the rerooted tree.

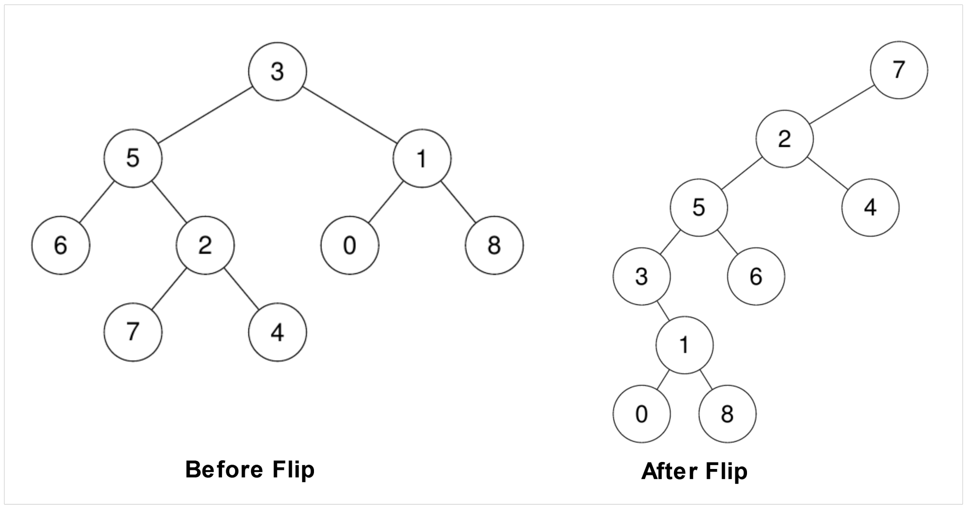
Note:

Ensure that your solution sets the

Node.parent

pointers correctly after rerooting or you will receive "Wrong Answer".

Example 1:



Input:

root = [3,5,1,6,2,0,8,null,null,7,4], leaf = 7

Output:

[7,2,null,5,4,3,6,null,null,null,1,null,null,0,8]

Example 2:

Input:

root = [3,5,1,6,2,0,8,null,null,7,4], leaf = 0

Output:

[0,1,null,3,8,5,null,null,null,6,2,null,null,7,4]

Constraints:

The number of nodes in the tree is in the range

[2, 100]

.

-10

9

<= Node.val <= 10

9

All

Node.val

are

unique

.

leaf

exist in the tree.

Code Snippets

C++:

```
/*
// Definition for a Node.
class Node {
public:
```

```
int val;
Node* left;
Node* right;
Node* parent;
};
*/

class Solution {
public:
Node* flipBinaryTree(Node* root, Node * leaf) {

}
};
```

Java:

```
/*
// Definition for a Node.
class Node {
public int val;
public Node left;
public Node right;
public Node parent;
};
*/

class Solution {
public Node flipBinaryTree(Node root, Node leaf) {

}
}
```

Python3:

```
"""
# Definition for a Node.
class Node:
def __init__(self, val):
self.val = val
self.left = None
self.right = None
self.parent = None
```

```
"""

class Solution:
def flipBinaryTree(self, root: 'Node', leaf: 'Node') -> 'Node':
```

Python:

```
"""
# Definition for a Node.
class Node:
def __init__(self, val):
self.val = val
self.left = None
self.right = None
self.parent = None
"""

class Solution(object):
def flipBinaryTree(self, root, leaf):
"""
:type node: Node
:rtype: Node
"""
```

JavaScript:

```
/**
* // Definition for a Node.
* function Node(val) {
* this.val = val;
* this.left = null;
* this.right = null;
* this.parent = null;
* };
*/

/**
* @param {Node} node
* @return {Node}
*/
var flipBinaryTree = function(root, leaf) {
};
```

C#:

```
/*
// Definition for a Node.
public class Node {
public int val;
public Node left;
public Node right;
public Node parent;
}
*/

public class Solution {
public Node FlipBinaryTree(Node root, Node leaf) {

}
}
```

Solutions

C++ Solution:

```
/*
* Problem: Change the Root of a Binary Tree
* Difficulty: Medium
* Tags: tree, search
*
* Approach: DFS or BFS traversal
* Time Complexity: O(n) where n is number of nodes
* Space Complexity: O(h) for recursion stack where h is height
*/

/*
// Definition for a Node.
class Node {
public:
int val;
Node* left;
Node* right;
Node* parent;
};
*/

class Solution {
public:
Node* flipBinaryTree(Node* root, Node * leaf) {

}
};
```

Java Solution:

```
/**
* Problem: Change the Root of a Binary Tree
* Difficulty: Medium
* Tags: tree, search
*
* Approach: DFS or BFS traversal
* Time Complexity: O(n) where n is number of nodes
* Space Complexity: O(h) for recursion stack where h is height
*/

/*
// Definition for a Node.
class Node {
public int val;
public Node left;
public Node right;
public Node parent;
};
*/

class Solution {
public Node flipBinaryTree(Node root, Node leaf) {

}
}
```

Python3 Solution:

```
"""
Problem: Change the Root of a Binary Tree
Difficulty: Medium
Tags: tree, search

Approach: DFS or BFS traversal
Time Complexity: O(n) where n is number of nodes
Space Complexity: O(h) for recursion stack where h is height
"""

"""
# Definition for a Node.
class Node:
def __init__(self, val):
self.val = val
self.left = None
self.right = None
self.parent = None
"""

class Solution:
def flipBinaryTree(self, root: 'Node', leaf: 'Node') -> 'Node':
# TODO: Implement optimized solution
pass
```

Python Solution:

```
"""
# Definition for a Node.
class Node:
def __init__(self, val):
self.val = val
self.left = None
self.right = None
self.parent = None
"""

class Solution(object):
def flipBinaryTree(self, root, leaf):
"""
:type node: Node
:rtype: Node
```

```
"""
```

JavaScript Solution:

```
/**
* Problem: Change the Root of a Binary Tree
* Difficulty: Medium
* Tags: tree, search
*
* Approach: DFS or BFS traversal
* Time Complexity: O(n) where n is number of nodes
* Space Complexity: O(h) for recursion stack where h is height
*/

/**
* // Definition for a Node.
* function Node(val) {
* this.val = val;
* this.left = null;
* this.right = null;
* this.parent = null;
* };
*/

/**
* @param {Node} node
* @return {Node}
*/
var flipBinaryTree = function(root, leaf) {

};
```

C# Solution:

```
/*
* Problem: Change the Root of a Binary Tree
* Difficulty: Medium
* Tags: tree, search
*
* Approach: DFS or BFS traversal
* Time Complexity: O(n) where n is number of nodes
* Space Complexity: O(h) for recursion stack where h is height
*/

/*
// Definition for a Node.
public class Node {
public int val;
public Node left;
public Node right;
public Node parent;
}
*/

public class Solution {
public Node FlipBinaryTree(Node root, Node leaf) {

}
}
```