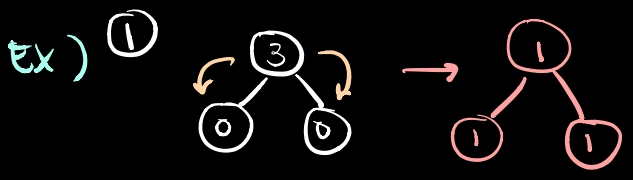
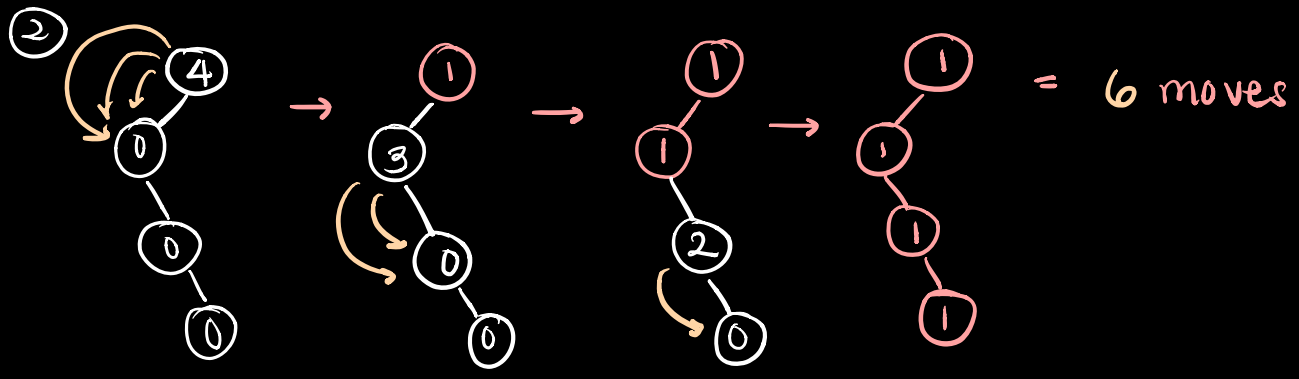
# Distribute Coins in a Binary Tree:

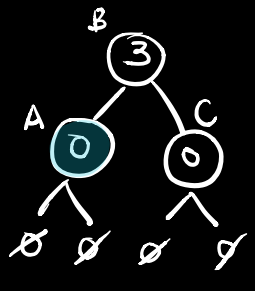
Given the root of a binary tree with N number of nodes, each node in the tree has node.val coins. There are N coins total.
In one move, we can choose 2 adjacent nodes and move a coin between them.
Return the number of moves required to distribute coins such that each node has one coin.

Ex) 


Since any kind of traversal would work here, I'm going with DFS.

For any node in the tree, we can calculate a separate value that tracks cumulative coin value. To track the total number of moves, we can use a global variable.

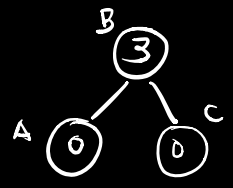



at node A, we have 0 coins. This guarantees a move needs to be made. Therefore, `node.value - 1` can be used to represent the "coin debt" as we traverse the tree. To also account for the child nodes, it becomes `node.value - 1 + leftDebt + rightDebt`.

when node.value is greater than 1, the debt from its children gets "paid".

To count the moves, the absolute value of left debt and the absolute value of right debt are added to the total moves.

Diagram:




Using post order traversal:

1) Node A
   coins = 0
   debt = 0 + 0 = 0
   moves = 0
   new debt = 0 - 1 - debt = -1

2) Node C
   coins = 0
   debt = 0 + 0 = 0
   moves = 0
   new debt = 0 - 1 - debt = -1

3) Node B
   coins = 0
   debt = abs(-1) + abs(-1) = 2
   moves = 2
   new debt = 3 - 1 - debt = 0

## Implementation

```
let totalMoves;

let dfs = function(root) {
    if (root === null) { return 0; }

    let leftDebt = dfs(root.left);
    let rightDebt = dfs(root.right);
    totalMoves += Math.abs(leftDebt) +
```

```
                                    Math.abs(rightDebt);
    return root.value - 1 + leftDebt + rightDebt;
}

let distributeCoins = function(root) {
    totalMoves = 0;
    dfs(root);
    return totalMoves;
}
```

## Complexity

Time: O(n) where n is the number of nodes in the tree. Every node is visited.

space: O(h) where h is the height of the tree. The recursive call stack reaches a maximum of the tree's height (memory frees up on the returns)