Difficulty: **Category:** Binary Trees **Successful Submissions:** 45,893+

Branch Sums

Write a function that takes in a Binary Tree and returns a list of its branch sums ordered from leftmost branch sum to rightmost branch sum.

A branch sum is the sum of all values in a Binary Tree branch. A Binary Tree branch is a path of nodes in a tree that starts at the root node and ends at any leaf node.

Each `BinaryTree` node has an integer `value`, a `left` child node, and a `right` child node. Children nodes can either be `BinaryTree` nodes themselves or `None` / `null`.

Sample Input

```
tree =     1
         /     \
        2       3
      /   \    /  \
     4     5  6    7
   /   \  /
  8    9 10
```

Sample Output

```
[15, 16, 18, 10, 11]
// 15 == 1 + 2 + 4 + 8
// 16 == 1 + 2 + 4 + 9
// 18 == 1 + 2 + 5 + 10
// 10 == 1 + 3 + 6
// 11 == 1 + 3 + 7
```

Hints

Hint 1

Try traversing the Binary Tree in a depth-first-search-like fashion.

Hint 2

Recursively traverse the Binary Tree in a depth-first-search-like fashion, and pass a running sum of the values of every previously-visited node to each node that you're traversing.

Hint 3

As you recursively traverse the tree, if you reach a leaf node (a node with no "left" or "right" Binary Tree nodes), add the relevant running sum that you've calculated to a list of sums (which you'll also have to pass to the recursive function). If you reach a node that isn't a leaf node, keep recursively traversing its children nodes, passing the correctly updated running sum to them.

Optimal Space & Time Complexity

O(n) time | O(n) space - where n is the number of nodes in the Binary Tree