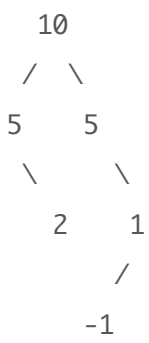
Daily Coding Problem #135

Problem

This question was asked by Apple.

Given a binary tree, find a minimum path sum from root to a leaf.

For example, the minimum path in this tree is [10, 5, 1, -1], which has sum 15.



Solution

This question can be solved using structural induction.

- Assuming that we can find the `min_sum_path` of `node.left` and `node.right`, how could we use these results for `node`?
- We can pick one of `min_sum_path(node.left)` `min_sum_path(node.right)`, whichever has the min sum, and add `node` to

the end of that list.

- For the base case, we simply return [] since no such path exists.

Since the above code returns the list in reversed order, we can create a wrapper method that can reverse this list to make it in proper order:

```
def min_sum_path(node):
    return list(reversed(_min_sum_path(node)))

def _min_sum_path(node):
    if node:
        left_list = _min_sum_path(node.left)
        right_list = _min_sum_path(node.right)

        min_list = min(left_list, right_list, key=lambda lst: sum(node.val for
node in lst))
        min_list.append(node)

        return min_list

    return []
```

This code takes O(N) time since it traverses the whole tree but takes O(K) memory where K is the height of the tree.