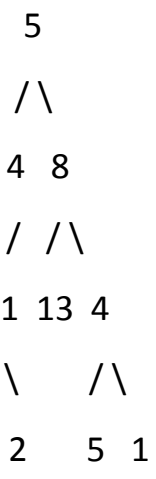
Path Sum II

Question: Given a binary tree and a sum, find all root-to-leaf paths where each path's sum equals the given sum.

For example:

Given the below binary tree and sum = 22,



return

[[5,4,11,2], [5,8,4,5]]

Solutions:

```
class TreeNode:
    def __init__(self, x):
        self.val = x
        self.left = None
        self.right = None


class Solution:
    # @param root, a tree node
    # @param sum, an integer
    # @return a list of lists of integers
    def pathSum(self, root, sum):
```

```
        solution = []
        self.pathSumRec(root, sum, 0, [], solution)
        return solution

    def pathSumRec(self, root, sum, tempSum, tempList, solution):
        if root == None:
            return
        tempList.append(root.val)
        tempSum += root.val
        if root.left == None and root.right == None:
            if tempSum == sum:
                solution.append(list(tempList))
        else:
            self.pathSumRec(root.left, sum, tempSum, tempList, solution)
            self.pathSumRec(root.right, sum, tempSum, tempList, solution)
        tempList.pop()

if __name__ == '__main__':
    BT, BT.right, BT.right.left, BT.left = TreeNode(1), TreeNode(2), TreeNode(3),
TreeNode(10)
    print ( Solution().pathSum(BT, 6) )
```