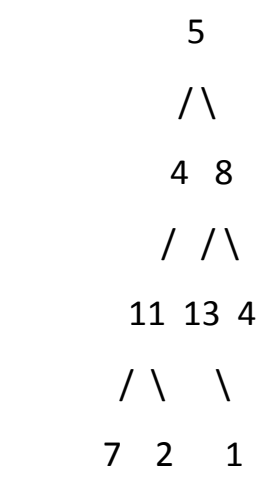
Path Sum

Question: Given a binary tree and a sum, determine if the tree has a root-to-leaf path such that adding up all the values along the path equals the given sum.

For example: Given the below binary tree and sum = 22,



return true, as there exist a root-to-leaf path 5->4->11->2 which sum is 22.

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 boolean
    def hasPathSum(self, root, sum):
        if root == None:
```

```
            # Empty tree will always result in False
            return False
        elif root.left == None and root.right == None:
            # Reach the leaf.
            return root.val == sum
        elif root.left == None:
            # Only has right child.
            return self.hasPathSum(root.right, sum-root.val)
        elif root.right == None:
            # Only has left child.
            return self.hasPathSum(root.left, sum-root.val)
        else:
            # Has two children.
            return self.hasPathSum(root.left, sum-root.val) or
self.hasPathSum(root.right, sum-root.val)


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