Binary Tree Maximum Path Sum

Question: Given a binary tree, find the maximum path sum.

The path may start and end at any node in the tree.

For example:

Given the below binary tree,

```
  1
 / \
2   3
```

Return 6.

Solutions:

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


class Solution:
    # @param root, a tree node
    # @return an integer
    def maxPathSum(self, root):
        self.maxValue = float("-inf")
        self.maxPathSumRec(root)
        return self.maxValue


    def maxPathSumRec(self, root):
```

```
        if root == None:
            return 0
        leftSum = self.maxPathSumRec(root.left)
        rightSum = self.maxPathSumRec(root.right)
        if leftSum<0 and rightSum<0:
            self.maxValue = max(self.maxValue, root.val)
            return root.val
        if leftSum>0 and rightSum>0:
            self.maxValue = max(self.maxValue, root.val+leftSum+rightSum)
        maxValueUp = max(leftSum, rightSum) +root.val
        self.maxValue = max(self.maxValue, maxValueUp)
        return maxValueUp

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