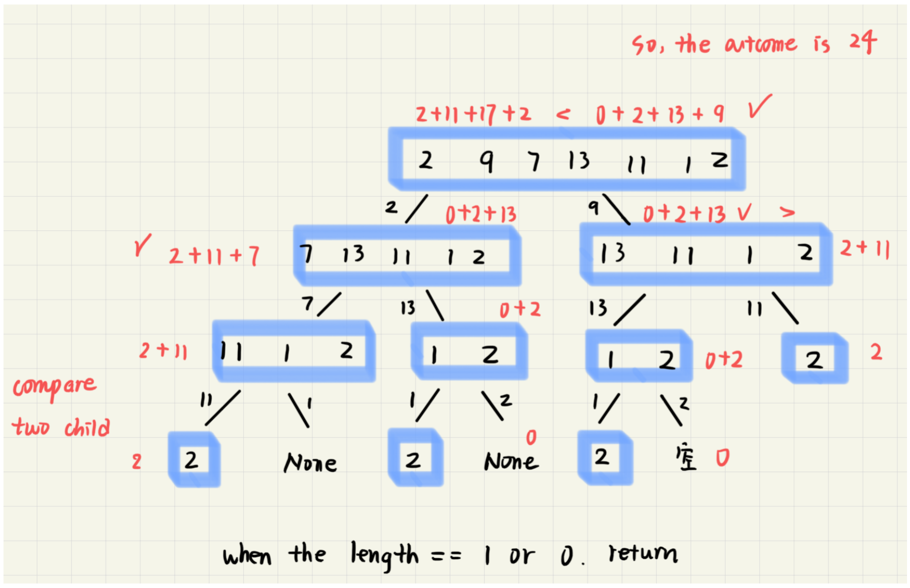
198. House Robber

My solution:



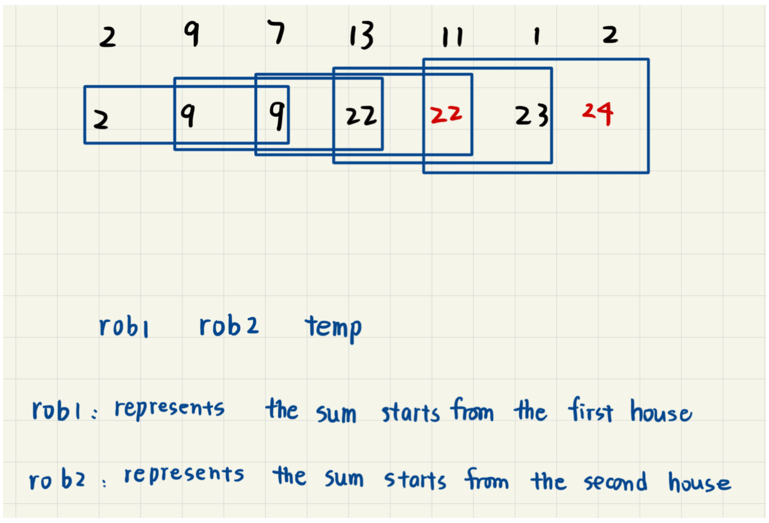
```python
class Solution:
    def rob(self, nums: List[int]) -> int:
        target = len(nums) // 2
        return self.compare(nums, target)

    def compare(self, nums, target, memo):
        if len(nums) == 1:
            return nums[0]
        if len(nums) == 0:
            return 0
        res = 0
        for i in range(target):
            left = nums[0] + self.compare(nums[2:], target)
            right = nums[1] + self.compare(nums[3:],target)
            res = max(left, right)

        return res

```

Time complexity is out of range. **Hashtable can't store list, so my memorization failed. sigh...**

Standard Solution:

https://youtu.be/73r3KWiEvyk

He uses iteration to solve this out.


2 9 7 13 11 1 2
2 9 9 22 22 23 24
rob1 rob2 temp
rob1 : represents the sum starts from the first house
rob2 : represents the sum starts from the second house

temp represents the max among uneven pick and even pick.

```python
class Solution:
    def rob(self, nums: List[int]) -> int:
        rob1, rob2 = 0, 0
        for n in nums:
            temp = max(n + rob1, rob2)
            rob1 = rob2
            rob2 = temp
        return rob2
```