213. House Robber II

You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed. All houses at this place are **arranged in a circle.** That means the first house is the neighbor of the last one. Meanwhile, adjacent houses have a security system connected, and **it will automatically contact the police if two adjacent houses were broken into on the same night**.

Given an integer array `nums` representing the amount of money of each house, return *the maximum amount of money you can rob tonight* ***without alerting the police***.

Example 1:

```
Input: nums = [2,3,2]
Output: 3
Explanation: You cannot rob house 1 (money = 2) and then rob house 3
(money = 2), because they are adjacent houses.
```

Example 2:

```
Input: nums = [1,2,3,1]
Output: 4
Explanation: Rob house 1 (money = 1) and then rob house 3 (money = 3).
Total amount you can rob = 1 + 3 = 4.
```

Example 3:

```
Input: nums = [1,2,3]
Output: 3
```

```
class Solution:
    def rob(self, nums: List[int]) -> int:
        n = len(nums)
        if n==1:
            return nums[0]
        if n==2:
            return max(nums)

        dp1 = [0]*n
        dp1[0] = nums[0]
        dp1[1] = max(nums[0],nums[1])
        for i in range(2,n-1):
            dp1[i] = max(dp1[i-1],nums[i]+dp1[i-2])
        dp2 = [0]*n
        dp2[1] = nums[1]
        dp2[2] = max(nums[1],nums[2])
        for i in range(3,n):
            dp2[i] = max(dp2[i-1],nums[i]+dp2[i-2])
        return max(max(dp1),max(dp2))
```