House Robber

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 a list of non-negative integers `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 = [0]
Output: 0
```

Constraints:

- `1 <= nums.length <= 100`
- `0 <= nums[i] <= 1000`

Solution 01

The following solution works if the houses are **NOT arranged in a circle**.

```
const list = [1,2,3,1];

const rob = (list) => {
    let pos = 0;
    let amount = 0;
    if (list.length == 0) return 0;
    else if (list.length <= 2) return list[0];

    while(pos < list.length) {
        let temp = list.splice(pos,1)[0];
        amount = amount + temp;
        pos = pos+1;
    }

    return amount;
};

console.log(rob(list));
```

Solution 02

The following solution works if the houses **ARE arranged in a circle**.

```
const list = [1,2,3,1];

const rob = (list) => {
    let pos = 0;
    let amount = 0;
    if (list.length == 0) return 0;
    else if (list.length <= 2) return 0;
    else if (list.length === 3) return list[1];

    while(pos < list.length) {
        let temp = list.splice(pos,1)[0];
        amount = amount + temp;
        pos = pos+1;
    }

    return amount;
};

console.log(rob(list));
```