你是一个专业的小偷，计划偷窃沿街的房屋。每间房内都藏有一定的现金，影响你偷窃的唯一制约因素就是相邻的房屋装有相互连通的防盗系统，**如果两间相邻的房屋在同一晚上被小偷闯入，系统会自动报警。**

给定一个代表每个房屋存放金额的非负整数数组，计算你**在不触动警报装置的情况下，**能够偷窃到的最高金额。

示例 1:

输入： [1,2,3,1]
输出： 4
解释： 偷窃 1 号房屋 (金额 = 1) ，然后偷窃 3 号房屋 (金额 = 3)。
偷窃到的最高金额 = 1 + 3 = 4 。

示例 2:

输入： [2,7,9,3,1]
输出： 12
解释： 偷窃 1 号房屋 (金额 = 2), 偷窃 3 号房屋 (金额 = 9), 接着偷窃 5 号房屋 (金额 = 1)。
偷窃到的最高金额 = 2 + 9 + 1 = 12 。

思路：这题是动态规划题，该题的切入点为小偷不能偷相邻家的钱财，那么小偷偷当前家获得的最大钱财等于他之前2，3家中最大的钱财与当前家的和（更远的被2，3家给吃了，比小于2，3家）只需要动态剔除前0，1，2家的最大钱财，后面的可以动态规划得到。

代码

```
class Solution {
public:
    int rob(vector<int>& nums) {
        int max=0;
        for(int i=0;i<nums.size();i++){
            if(i==0){
                max=nums[i];
            }else if(i==1){
                max=nums[i]>max?nums[i]:max;
            }else if(i==2){
                nums[i]=nums[i]+nums[i-2];
                if(nums[i]>max){
                    max=nums[i];
                }
            }else{
                nums[i]=nums[i]+(nums[i-2]>nums[i-3]?nums[i-2]:nums[i-3]);
                if(nums[i]>max){
                    max=nums[i];
                }
            }
```

```
        }
        return max;
    }
};
```