一、题目说明

题目198. House Robber，给一列正整数表示每个房间存的金币，不能连续抢2个房间，计算可以得到的最大金币。

二、我的解答

这个题目，我列举了n=1,2,3，...5的情况，没有找到规律。后面看了解答知道了：

```
dp[i+1]= max(dp[i-2]+nums[i],dp[i-1])
```

代码如下：

```
class Solution{
    public:
        int dfs(vector<int>& nums,int n){
            if(n==0) return 0;
            if(n==1) return nums[0];
            return max(dfs(nums,n-2)+nums[n-1],dfs(nums,n-1));
        }
        int rob(vector<int>& nums){
            return dfs(nums,nums.size());
        }
};
```

遗憾的是，超时：**Time Limit Exceeded**，通过map，裁剪如下：

```
class Solution{
    public:
        int dfs(vector<int>& nums,int n){
            if(n==0) return 0;
            if(n==1) return nums[0];
            if(ump.count(n)>0){
                return ump[n];
            }
            int result = max(dfs(nums,n-2)+nums[n-1],dfs(nums,n-1));
            ump[n] = result;
            return result;
        }
        int rob(vector<int>& nums){

            return dfs(nums,nums.size());
        }
    private:
        unordered_map<int,int> ump;
};
```

```
Runtime: 4 ms, faster than 58.41% of C++ online submissions for House Robber.
Memory Usage: 9.2 MB, less than 5.66% of C++ online submissions for House
Robber.
```

三、优化措施

用dp方法，代码如下：

```
class Solution{
    public:
        int rob(vector<int>& nums){
            //dp[i]表示抢第i所房子的最大值，
            //dp[i+1]= max(dp[i-2]+nums[i],dp[i-1] )
            if(nums.size()<1) return 0;
            if(nums.size() == 1) return nums[0];
            if(nums.size() == 2) return max(nums[0],nums[1]);

            vector<int> dp(nums.size()+1,0);
            dp[0] = nums[0];
            dp[1] = max(nums[0],nums[1]);

            for(int i=2;i<nums.size();i++){
                dp[i] = max(dp[i-2]+nums[i],dp[i-1]);
            }
            return dp[nums.size()-1];
        }
};
```

性能如下：

```
Runtime: 0 ms, faster than 100.00% of C++ online submissions for House Robber.
Memory Usage: 8.6 MB, less than 77.36% of C++ online submissions for House
Robber.
```