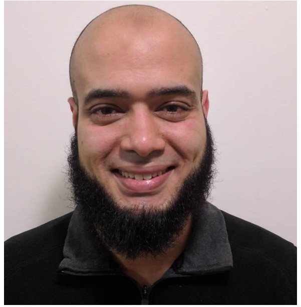
Algorithms

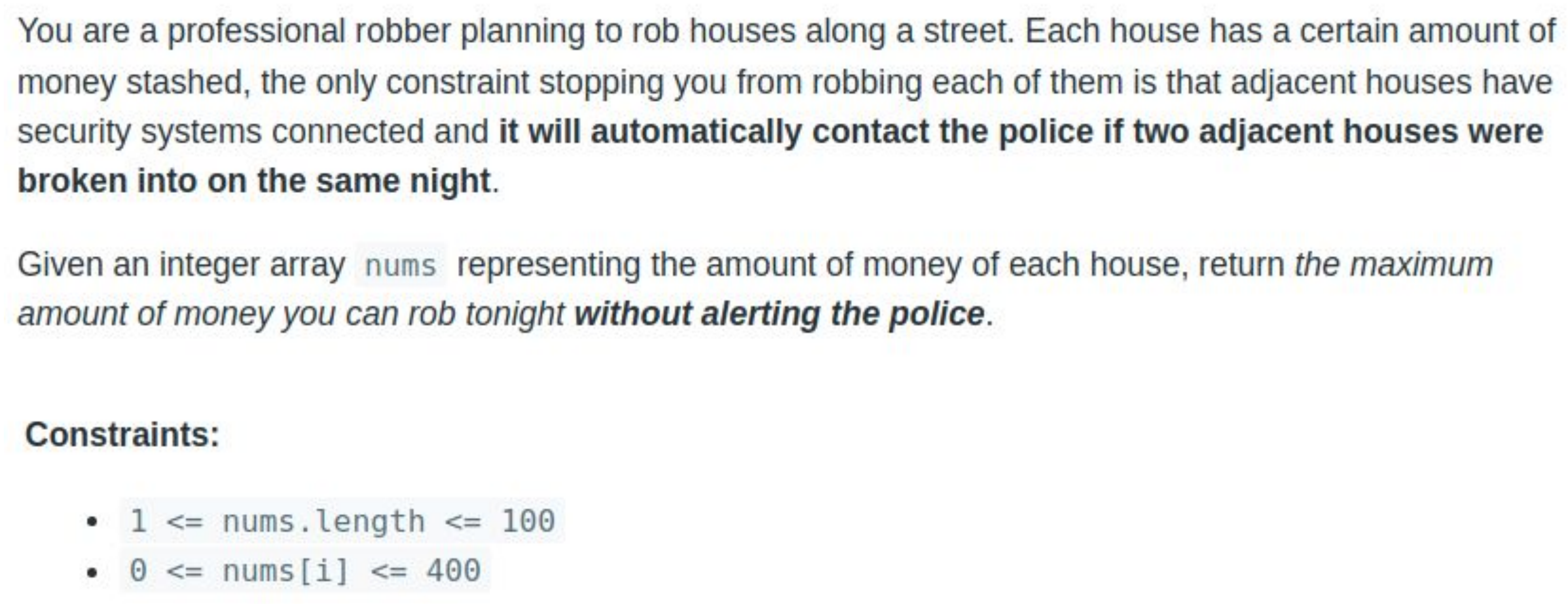
DP homework 2

Mostafa S. Ibrahim
Teaching, Training and Coaching for more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Problem #1: LeetCode 198 - House Robber

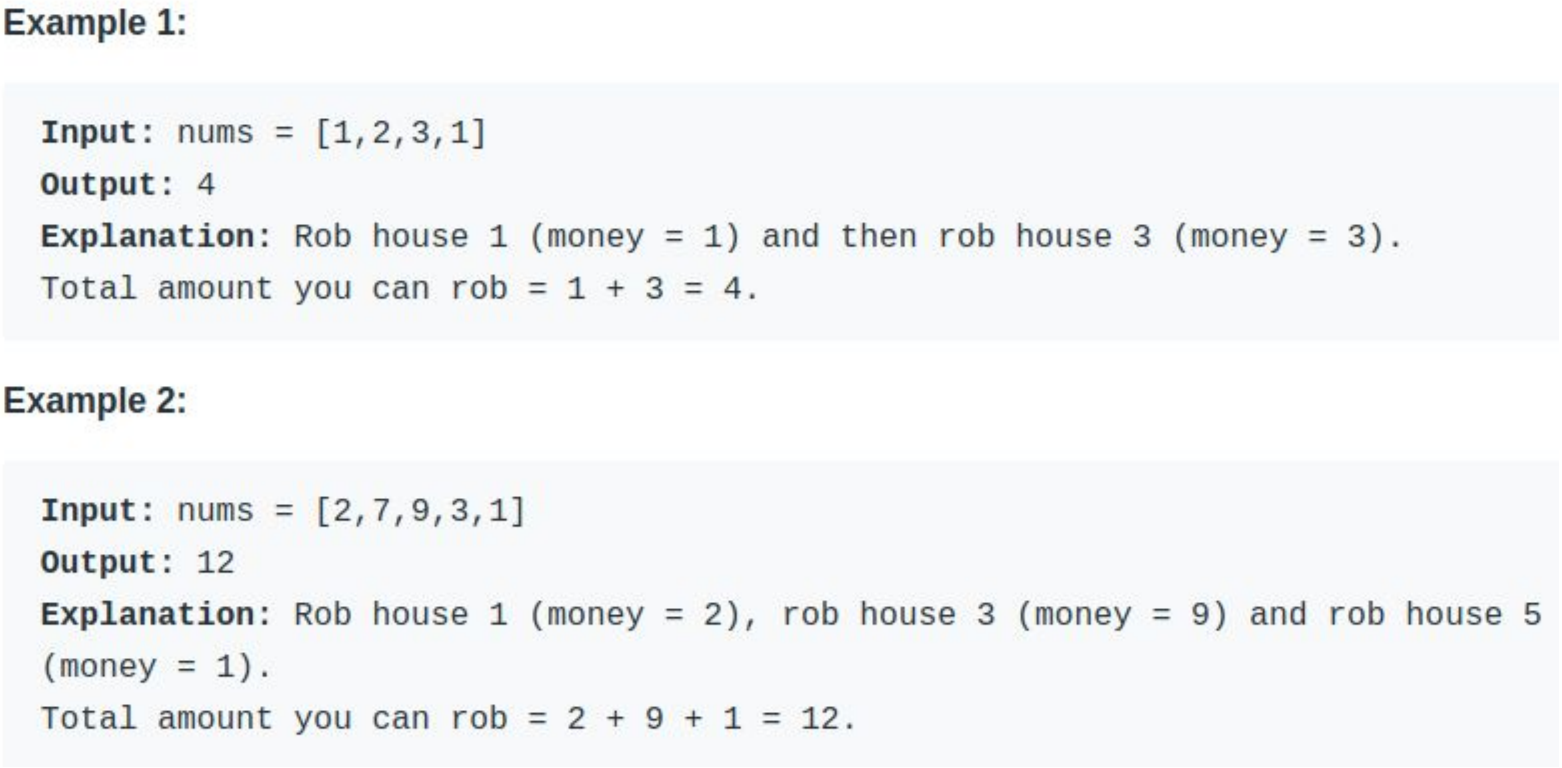


You are a professional robber planning to rob houses along a street. Each house has a certain amount of money stashed, the only constraint stopping you from robbing each of them is that adjacent houses have security systems connected and **it will automatically contact the police if two adjacent houses 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***.

Constraints:

- `1 <= nums.length <= 100`
- `0 <= nums[i] <= 400`



Example 1:

```
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 2:

```
Input: nums = [2,7,9,3,1]
Output: 12
Explanation: Rob house 1 (money = 2), rob house 3 (money = 9) and rob house 5
(money = 1).
Total amount you can rob = 2 + 9 + 1 = 12.
```

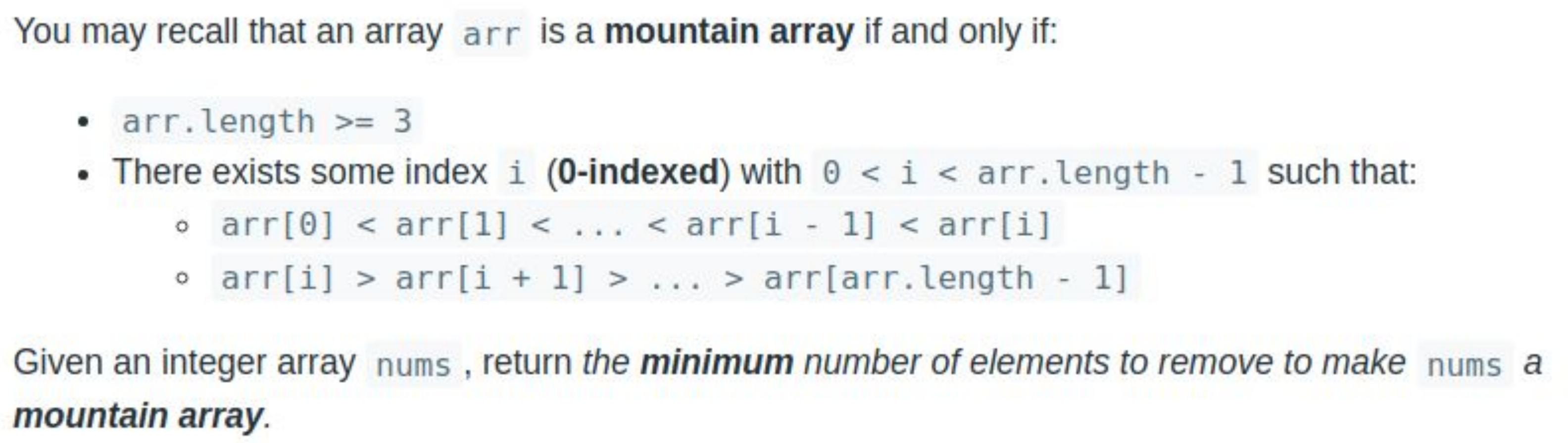
Problem #2: LeetCode 309 - Best Time to Buy and Sell Stock with Cooldown

- Rephrasing!
- You are given an array prices where prices[i] is the price of a given stock on the i^{th} day. You can buy and sell according to these constraints:
    - You start with no stocks.
    - At anytime you either have no stocks or a **single** stock
    - Logically you can’t sell a stock if you don’t have one
    - After you sell your stock, you cannot buy stock on the next day (cooldown day)
        - In other words, to buy on day x, you must do nothing on day x-1
- Goal: Find the **maximum profit** you can achieve
- Constraints
    - 1 <= prices.length <= 5000
    - 0 <= prices[i] <= 1000

Examples

- Prices: [**1, 2**, 3, **0, 2**] ⇒ Output: 3
    - Transactions = [buy, sell, cooldown, buy, sell]
        - Buy for $1: now we lost $1
        - Sell it for $2: we gain $2. Current profit is 2-1 = $1
        - Do nothing
        - Buy for $0. Still profit is 1 and we have a stock
        - Sell it for $2: we gain $2. Current profit is 2 + 1 = $3
- Prices: [10] ⇒ Output: 0 [better do nothing]
- Prices: [**1**, 2, 3, 4, **5**] ⇒ Output: 4
- Prices: [5, 4, 3, 2, 1] ⇒ Output: 0
- Prices: [**1**, 10, **15**] ⇒ Output: 14
- Prices: [3, **0, 15**, 20, **1, 12**] ⇒ Output: 26

Problem #3: LeetCode 1671 - Minimum Number of Removals to Make Mountain Array

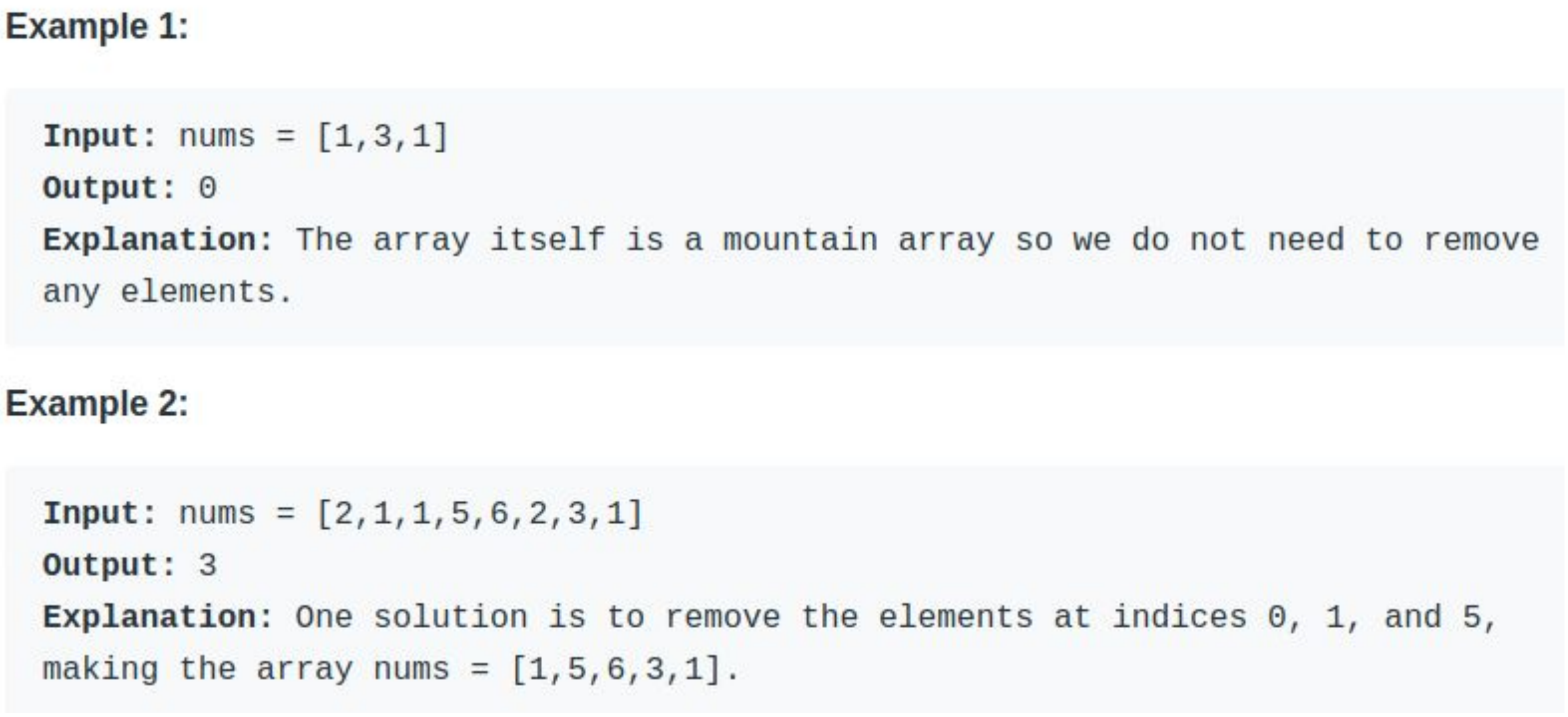


You may recall that an array `arr` is a **mountain array** if and only if:

- `arr.length >= 3`
- There exists some index `i` (**0-indexed**) with `0 < i < arr.length - 1` such that:
  - `arr[0] < arr[1] < ... < arr[i - 1] < arr[i]`
  - `arr[i] > arr[i + 1] > ... > arr[arr.length - 1]`

Given an integer array `nums`, return *the **minimum** number of elements to remove to make* `nums` ***a mountain array***.

- 3 <= nums.length <= 1000 $1 \le nums[i] \le 10^9$
- **Hint**: Find observations to relate the problem to LIS variant



Example 1:

```
Input: nums = [1,3,1]
Output: 0
Explanation: The array itself is a mountain array so we do not need to remove
any elements.
```

Example 2:

```
Input: nums = [2,1,1,5,6,2,3,1]
Output: 3
Explanation: One solution is to remove the elements at indices 0, 1, and 5,
making the array nums = [1,5,6,3,1].
```

“Acquire knowledge and impart it to the people.”

“Seek knowledge from the Cradle to the Grave.”