

2560. House Robber IV

My Submissions (/contest/weekly-contest-331/problems/house-robber-iv/submissions/)

Back to Contest (/contest/weekly-contest-331/)

There are several consecutive houses along a street, each of which has some money inside. There is also a robber, who wants to steal money from the homes, but he **refuses to steal from adjacent homes**.

The **capability** of the robber is the maximum amount of money he steals from one house of all the houses he robbed.

You are given an integer array `nums` representing how much money is stashed in each house. More formally, the i^{th} house from the left has `nums[i]` dollars.

You are also given an integer `k`, representing the **minimum** number of houses the robber will steal from. It is always possible to steal at least `k` houses.

Return the **minimum** capability of the robber out of all the possible ways to steal at least `k` houses.

User Accepted:	1535
User Tried:	4353
Total Accepted:	1608
Total Submissions:	9573
Difficulty:	Medium

Example 1:

Input: `nums = [2,3,5,9], k = 2`
Output: `5`
Explanation:
There are three ways to rob at least 2 houses:
– Rob the houses at indices 0 and 2. Capability is `max(nums[0], nums[2]) = 5`.
– Rob the houses at indices 0 and 3. Capability is `max(nums[0], nums[3]) = 9`.
– Rob the houses at indices 1 and 3. Capability is `max(nums[1], nums[3]) = 9`.
Therefore, we return `min(5, 9, 9) = 5`.

Example 2:

Input: `nums = [2,7,9,3,1], k = 2`
Output: `2`
Explanation: There are 7 ways to rob the houses. The way which leads to minimum capability is to rob the house at index 0 and index 4. The minimum capability is `max(nums[0], nums[4]) = 2`.

Constraints:

- `1 <= nums.length <= 105`
- `1 <= nums[i] <= 109`
- `1 <= k <= (nums.length + 1)/2`

Discuss (<https://leetcode.com/problems/house-robber-iv/discuss>)

JavaScript

📄

↺

⚙️


```
1 let a, k;
2 const minCapability = (A, K) => {
3   a = A, k = K;
4   return BinarySearch(0, Math.max(...a))
5 };
6
7 const BinarySearch = (low, high) => {
8   while (low <= high) {
9     let mid = low + parseInt((high - low) / 2);
10    if (possible(mid)) {
11      low = mid + 1;
12    } else {
13      high = mid - 1;
14    }
15  }
16  return low;
17 };
```

```
18
19 ▾ const possible = (v) => {
20     let cnt = 0, isPick = false;
21 ▾   for (const x of a) {
22 ▾       if (x <= v && !isPick) {
23           cnt++;
24           isPick = true;
25 ▾       } else {
26           isPick = false;
27       }
28   }
29   return cnt < k;
30 };
```

☐ Custom Testcase[Use Example Testcases](#)[Run](#)[Submit](#)**Submission Result: Accepted** (</submissions/detail/891863161/>) ?[More Details >](/submissions/detail/891863161/)

Share your acceptance!

Copyright © 2023 LeetCode

[Help Center \(/support/\)](/support/) | [Jobs \(/jobs\)](/jobs/) | [Bug Bounty \(/bugbounty\)](/bugbounty/) | [Online Interview \(/interview/\)](/interview/) | [Students \(/student\)](/student/) | [Terms \(/terms\)](/terms/) | [Privacy Policy \(/privacy\)](/privacy/) [United States \(/region\)](/region/)