Discuss(/discuss/)

Æ⊟oc(https://deeteade;com/discuss/general-Storediscussion/655704/)

☆ Premium (/subscribe? ref=nb npl)





# 5556. Furthest Building You Can Reach

My Submissions (/contest/weekly-contest-213/problems/furthest-building-you-can-reach/submissions/)

Back to Contest (/contest/weekly-contest-213/)

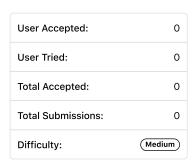
You are given an integer array heights representing the heights of buildings, some bricks, and some ladders.

You start your journey from building 0 and move to the next building by possibly using bricks or ladders.

While moving from building i to building i+1 (0-indexed),

- If the current building's height is greater than or equal to the next building's height, you do not need a ladder or
- If the current building's height is less than the next building's height, you can either use one ladder or (h[i+1] - h[i]) bricks.

Return the furthest building index (0-indexed) you can reach if you use the given ladders and bricks optimally.



## Example 1:



Input: heights = [4,2,7,6,9,14,12], bricks = 5, ladders = 1 Output: 4

Explanation: Starting at building 0, you can follow these steps: - Go to building 1 without using ladders nor bricks since 4 >= 2.

- Go to building 2 using 5 bricks. You must use either bricks or ladders because 2 < 7.
- Go to building 3 without using ladders nor bricks since 7 >= 6.
- Go to building 4 using your only ladder. You must use either bricks or ladders because 6 < 9.

It is impossible to go beyond building 4 because you do not have any more bricks or ladders.

### Example 2:

Input: heights = [4,12,2,7,3,18,20,3,19], bricks = 10, ladders = 2 Output: 7

## Example 3:

```
Input: heights = [14,3,19,3], bricks = 17, ladders = 0
Output: 3
```

#### **Constraints:**

- 1 <= heights.length <= 10<sup>5</sup>
   1 <= heights[i] <= 10<sup>6</sup>
   0 <= bricks <= 10<sup>9</sup>
   0 <= ladders <= heights.length</li>
- JavaScript Ø C 1 - /\*\* \* @param {number[]} heights 2 \* @param {number} bricks 3 \* @param {number} ladders 4 5 \* @return {number} 6 7 ▼ const furthestBuilding = (heights, bricks, ladders) => { 8 let n = heights.length; let diff = []; 9 for (let i = 0; i + 1 < n; i++) { 10 ▼ 11 diff.push(heights[i + 1] - heights[i]); 12 13 let tmp = [...diff].sort((a, b) => b - a).filter(x => x > 0); 14 let cnt = 0; 15 ▼ for (let i = 0; i < diff.length; i++) { 16 ▼ if (diff[i] <= 0) { 17 cnt++; 18 continue; 19 if ((tmp.indexOf(diff[i]) + 1) <= ladders) {</pre> 20 • if (ladders > 0) { 21 ▼ ladders--; 22 23 cnt++; 24 25 ▼ } else { if (bricks - diff[i] >= 0) { 26 ▼ 27 bricks -= diff[i]; 28 cnt++; 29 ▼ } else { if (ladders > 0) { 30 ▼ 31 ladders--; 32 cnt++; 33 ▼ } else { 34 break; 35 36 } 37 } 38 39 return cnt; }; 40

☐ Custom Testcase Use Example Testcases

Submission Result: Accepted (/submissions/detail/415449594/) 2

More Details > (/submissions/detail/415449594/)

Run

Submit

Share your acceptance!

Copyright © 2020 LeetCode Help Center (/support) | Jobs (/jobs) | Bug Bounty (/bugbounty) | Students (/student) | Terms (/terms) | Privacy Policy (/privacy)