Previous Next

755. Pour Water

Dec. 30, 2017 | 15.6K views

Average Rating: 1.95 (37 votes)

We are given an elevation map, `heights[i]` representing the height of the terrain at that index. The width at each index is 1. After `V` units of water fall at index `K`, how much water is at each index?

Water first drops at index `K` and rests on top of the highest terrain or water at that index. Then, it flows according to the following rules:

- If the droplet would eventually fall by moving left, then move left.
- Otherwise, if the droplet would eventually fall by moving right, then move right.
- Otherwise, rise at it's current position.

Here, "eventually fall" means that the droplet will eventually be at a lower level if it moves in that direction. Also, "level" means the height of the terrain plus any water in that column.

We can assume there's infinitely high terrain on the two sides out of bounds of the array. Also, there could not be partial water being spread out evenly on more than 1 grid block - each unit of water has to be in exactly one block.

Example 1:

```
Input: heights = [2,1,1,2,1,2,2], V = 4, K = 3
Output: [2,2,2,3,2,2,2]
Explanation:
#       #
#       #
##  # ###
#########
 0123456    <- index

The first drop of water lands at index K = 3:

#       #
#   w   #
##  # ###
#########
 0123456

When moving left or right, the water can only move to the same level or a lower level.
(By level, we mean the total height of the terrain plus any water in that column.)
Since moving left will eventually make it fall, it moves left.
(A droplet "made to fall" means go to a lower height than it was at previously.)

#       #
#       #
## w# ###
#########
 0123456

Since moving left will not make it fall, it stays in place.  The next droplet falls:

#       #
#   w   #
## w# ###
#########
 0123456

Since the new droplet moving left will eventually make it fall, it moves left.
Notice that the droplet still preferred to move left,
even though it could move right (and moving right makes it fall quicker.)

#       #
#  w    #
## w# ###
#########
 0123456

#       #
#       #
##ww# ###
#########
 0123456

After those steps, the third droplet falls.
Since moving left would not eventually make it fall, it tries to move right.
Since moving right would eventually make it fall, it moves right.

#       #
#   w   #
##ww# ###
#########
 0123456

#       #
#       #
##ww#w###
#########
 0123456

Finally, the fourth droplet falls.
Since moving left would not eventually make it fall, it tries to move right.
Since moving right would not eventually make it fall, it stays in place:

#       #
#   w   #
##ww#w###
#########
 0123456

The final answer is [2,2,2,3,2,2,2]:

    #
 #######
 #######
 0123456
```

Example 2:

```
Input: heights = [1,2,3,4], V = 2, K = 2
Output: [2,3,3,4]
Explanation:
The last droplet settles at index 1, since moving further left would not cause it to e
```

Example 3:

```
Input: heights = [3,1,3], V = 5, K = 1
Output: [4,4,4]
```

Note:

1. `heights` will have length in `[1, 100]` and contain integers in `[0, 99]`.
2. `V` will be in range `[0, 2000]`.
3. `K` will be in range `[0, heights.length - 1]`.

Approach #1: Simulation [Accepted]

Intuition and Algorithm

We attempt to perform the steps directly as described.

First, notice that an index with terrain or with water is indistinguishable with respect to the flow of water. Thus, we can model `heights` as the total level of terrain and water directly as we perform our simulation.

When a droplet falls, we should check if it is possible for it to fall left. For our left pointer `i = K`, if `i - 1` is in bounds and `heights[i - 1] <= heights[i]`, the water will fall to a candidate block in that direction. We keep track of every time we actually fall at index `best`. If we "eventually fall" (`best != K`) as described in the problem statement, then we will drop the water there.

Otherwise, (if moving left will not cause the droplet to eventually fall), we can perform a similar check for `i = K` going right, and otherwise the droplet stays in place.

For convenience, we will name the initial array `H = heights`.

Java | Python | Copy

```python
class Solution(object):
    def pourWater(self, H, V, K):
        for _ in xrange(V):
            for d in (-1, 1):
                i = best = K
                while 0 <= i+d < len(H) and H[i+d] <= H[i]:
                    if H[i+d] < H[i]: best = i+d
                    i += d
                if best != K:
                    H[best] += 1
                    break
            else:
                H[K] += 1
        return H
```

Complexity Analysis

- Time Complexity: $O(V * N)$, where N is the length of `heights`. For each of V droplets, our while loop might iterate N times.

- Space Complexity: $O(1)$ in additional space complexity.

Analysis written by: @awice.

Rate this article:

Previous Next

Comments: 10 Sort By

Type comment here... (Markdown is supported)

Preview Post

wenhaocreepy 126 October 11, 2018 1:32 AM

Terrible solution

48 Share Reply

SHOW 2 REPLIES

rahulkun 375 March 10, 2019 11:26 PM

this problem is ambiguous.. reading from description and three examples it seems that ask is to find the left/right index that contains value lower than heights[K] and stop any time height[index]>heights[K]. Whereas, the test cases are meant to check whether program find the leftmost/rightmost position while constrained by heights[index]<=heights[K].

9 Share Reply

walrussgt 7 February 4, 2019 9:42 AM

```
class Solution {
    public int[] pourWater(int[] heights, int V, int K) {
        for (int i = 0; i < V; i++) {
            int pos = getFirstLowerFromLeft(heights, K);
```

Read More

3 Share Reply

SHOW 1 REPLY

noname_minion 9 June 17, 2018 11:57 PM

Seriously, people think this is a solution? In an interview, if your code has any sign of "goto", interviewer will ask you to remove it at best scenario. At worst scenario, you will get a vote of failure immediately.

5 Share Reply

SHOW 3 REPLIES

lijingyabeyond 18 March 15, 2020 5:58 PM

the problem statement is so difficult to understand...

1 Share Reply

foob 6 July 2, 2018 8:25 AM

To some extent I agree with noname.minion - checking in code like this to an actual real world project won't be good for the career or ones colleagues who have to maintain it. But that stated, LeetC is not all about that. It's about having fun and doing what you want.

1 Share Reply

wuyujack 0 May 16, 2019 10:57 AM

simple to read java solution:

```
    public int[] pourWater(int[] heights, int v, int k) {
      while (v > 0) {
        int left = k;
```

Read More

0 Share Reply

SHOW 1 REPLY

commando22 0 April 21, 2019 7:36 AM

//Simple C++ Solution Beats 100% Time and Memory
class Solution {
public:

```
void move(int cur_index, vector<int>& heights, vector<int>& wall){
```

Read More

0 Share Reply

lch04 8 March 7, 2018 12:26 PM

Code is hard to read...what's "droplet: " at line #3?

0 Share Reply

wzrthhj 131 August 5, 2018 9:26 AM

Never use break statement!

-4 Share Reply

SHOW 2 REPLIES