Open Feedback Dialog

codility

Training center

Check out Codility training tasks

Demo ticket

Session

ID: demoCR49Z6-R5W
Time limit: 120 min.

Status: closed

Created on: 2014-03-20 01:33 UTC
Started on: 2014-03-20 01:33 UTC
Finished on: 2014-03-20 01:41 UTC

Tasks in test

1 | {} ~~StoneWall~~

Task score

100%

Test score

100%

100 out of 100 points

MEDIUM

1. StoneWall

Cover "Manhattan skyline" using the minimum number of rectangles.

score: 100 of 100

Task description

Solution to this task can be found at our blog.
You are going to build a stone wall. The wall should be straight and N meters long, and its thickness should be constant; however, it should have different heights in different places. The height of the wall is specified by a zero-indexed array H of N positive integers. H[I] is the height of the wall from I to I+1 meters to the right of its left end. In particular, H[0] is the height of the wall's left end and H[N−1] is the height of the wall's right end.
The wall should be built of cuboid stone blocks (that is, all sides of such blocks are rectangular). Your task is to compute the minimum number of blocks needed to build the wall.
Write a function:

```
def solution(H)
```

that, given a zero-indexed array H of N positive integers specifying the height of the wall, returns the minimum number of blocks needed to build it.
For example, given array H containing N = 9 integers:

```
H[0] = 8    H[1] = 8    H[2] = 5
H[3] = 7    H[4] = 9    H[5] = 8
H[6] = 7    H[7] = 4    H[8] = 8
```

the function should return 7. The figure shows one possible arrangement of seven blocks.

Assume that:

- N is an integer within the range [1..100,000];
- each element of array H is an integer within the range [1..1,000,000,000].

Complexity:

- expected worst-case time complexity is O(N);
- expected worst-case space complexity is O(N), beyond input storage (not counting the storage required for input arguments).

Solution

Programming language used: Python

Total time used: 8 minutes

Effective time used: 8 minutes

Notes: correct functionality and scalability

Task timeline

01:33:55 01:41:51

Code: 01:41:51 UTC, py, final, score: **100.00**

```
01. def solution(H):
02.     i = 0
03.     stack = []
04.     count = 0
05.     while i < len(H):
06.         if not stack:
07.             stack.append(i)
08.             i += 1
09.         else:
10.             if H[stack[-1]] < H[i]:
11.                 stack.append(i)
12.                 i += 1
13.             elif H[stack[-1]] > H[i]:
14.                 stack.pop()
15.                 count += 1
16.             else:
17.                 i += 1
18.
19.     return count + len(stack)
```

Analysis

Elements of input arrays can be modified.

Detected time complexity:
O(N)

test	time	result
example	0.050 s.	**OK**
simple1	0.050 s.	**OK**
simple2	0.050 s.	**OK**
simple3	0.050 s.	**OK**
simple4	0.050 s.	**OK**
boundary_cases	0.050 s.	**OK**
medium1	0.050 s.	**OK**
medium2	0.050 s.	**OK**
medium3	0.050 s.	**OK**
medium4	0.050 s.	**OK**
large_piramid	0.290 s.	**OK**
large_increasing_decreasing	0.330 s.	**OK**
large_up_to_20	0.310 s.	**OK**
large_up_to_100	0.320 s.	**OK**
large_max	0.350 s.	**OK**

Training center