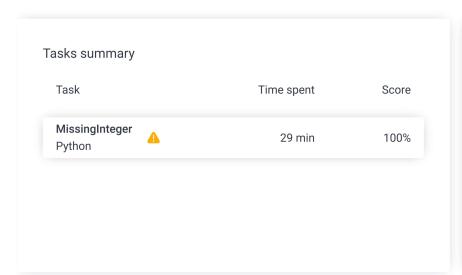
Codility_

Candidate Report: Anonymous

Test Name:

Summary Timeline

Check out Codility training tasks





Tasks Details

1. MissingInteger

Find the smallest positive integer that does not occur in a given

sequence.

Task Score

100%

Correctness

Performance

100%

100%

Task description

This is a demo task.

Write a function:

def solution(A)

that, given an array A of N integers, returns the smallest positive integer (greater than 0) that does not occur in A.

For example, given A = [1, 3, 6, 4, 1, 2], the function should return 5.

Given A = [1, 2, 3], the function should return 4.

Given A = [-1, -3], the function should return 1.

Write an efficient algorithm for the following assumptions:

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

Copyright 2009–2021 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

Solution Programming language used: Python Total time used: 29 minutes Effective time used: 29 minutes Notes: not defined yet Task timeline a 11:57:38 12:26:28 Code: 12:26:28 UTC, py, show code in pop-up final, score: 100 # you can write to stdout for debugging purposes, 2 # print("this is a debug message") 3

```
# <釐清問題>
 5
    # - 找到最小的正整數,且該正整數未出現於陣列中
 6
7
    #
 8
        A = [1, 3, 6, 4, 1, 2], the function should re
9
    # <其它限制>
10
    \# Given A = [1, 2, 3], the function should return
11
12
     # Given A = [-1, -3], the function should return 1
    # N is an integer within the range [1..100,000];
13
    # each element of array A is an integer within the
15
16
     def solution(A):
         return findMinDisappearInteger(A)
17
18
19
     def findMinDisappearInteger(array) -> int:
20
21
         non_duplicated_array = list(set(array))
22
23
         sorted_array = sorted(non_duplicated_array)
24
25
        cnt = 1
         for element in sorted_array:
26
27
            if element < 1:</pre>
28
                continue
             else:
                if cnt == element:
30
31
                    cnt += 1
32
                 else:
33
                     break
         return cnt
34
35
36
37
38
39
40
41
42
```

Analysis summary

The solution obtained perfect score.

Analysis

 $\begin{array}{c} \text{O(N) or} \\ \text{Oetected time complexity:} & \text{O(N *} \\ \text{log(N))} \end{array}$

collapse all		Example tests	
•	example1 first example test	√ OK	
1.	0.036 s OK		
•	example2 second example tes	√ OK t	
1.	0.040 s OK		
•	example3 third example test	√ OK	
1.	0.040 s OK		
colla	pse all	Correctness tests	

ľ		5 - 5 - 5 - 5 - 5 - 5 - 5 - 5 - 5 - 5 -	
	•	extreme_single a single element	✓ OK
	1.	0.036 s OK	
	2.	0.036 s OK	
	3.	0.036 s OK	
	4.	0.036 s OK	
ŀ	•	simple	✓ OK
		simple test	
	1.	0.036 s OK	
	2.	0.040 s OK	
	3.	0.036 s OK	
	•	extreme_min_max_value minimal and maximal values	✓ OK
	1.	0.040 s OK	
	2.	0.036 s OK	
	•	positive_only	✓ OK
		shuffled sequence of 0100 and then 102200	
	1.	0.040 s OK	
	2.	0.040 s OK	
	▼	negative_only shuffled sequence -1001	✓ 0K
	1.	0.036 s OK	
	colla	pse all Performance t	ests
	•	medium	√ OK
		chaotic sequences length=10005 (with minus)	
	1.	0.048 s OK	
	2.	0.048 s OK	
	3.	0.052 s OK	
	•	large_1	✓ OK
		chaotic + sequence 1, 2,, 40000 (without minus)	
	1.	0.136 s OK	
	•	large_2	√ OK
	·	shuffled sequence 1, 2,, 100000 (without minus)	· on
	1.	0.164 s OK	
	2.	0.148 s OK	
	•	large_3	√ OK
		chaotic + many -1, 1, 2, 3 (with minus)	
	1.	0.140 s OK	

The PDF version of this report that may be downloaded on top of this site may contain sensitive data including personal information. For security purposes, we recommend you remove it from your system once reviewed.