Codility_

CodeCheck Report: training8Y5B2B-N8E

Test Name:

Check out Codility training tasks

Summary | Timeline

Tasks summary

Task	Time spent	Score
MissingInteger ⚠ Python	7 min	100%

Total score

Tasks Details

Medium

1. **MissingInteger**
Find the smallest positive integer that does not occur in a given sequence.

Task Score	Correctness	Performance
100%	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.

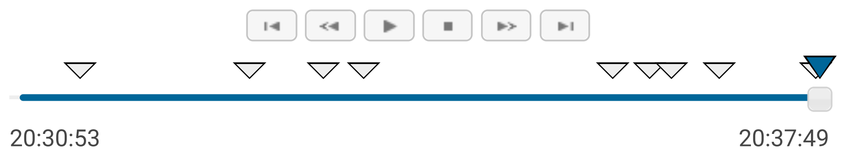
Write an **efficient** algorithm for the following assumptions:

- 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].

Solution

Programming language used:	Python
Total time used:	7 minutes
Effective time used:	7 minutes
Notes:	*not defined yet*

Task timeline



Code: 20:37:48 UTC, py, final, score: **100**

show code in pop-up

```
1  # you can write to stdout for debugging purposes, e.g.
2  # print("this is a debug message")
3
4  def solution(A):
5      # write your code in Python 3.6
6      if max(A)<1:
7          return 1
8      else:
9          A.sort()
10         lastnegative = -1
11         for i in range(len(A)):
12             if A[i]>0:
13                 if i == lastnegative+1:
14                     if A[i]>1:
15                         return 1
16                 else:
17                     if A[i]-A[i-1] > 1:
18                         return A[i-1]+1
19             else:
20                 lastnegative = i
21         return A[len(A)-1]+1
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: **O(N) or O(N * log(N))**

expand all	Example tests	
▶	example1 first example test	✔ OK
▶	example2 second example test	✔ OK
▶	example3 third example test	✔ OK
expand all	**Correctness tests**	
▶	extreme_single a single element	✔ OK
▶	simple simple test	✔ OK
▶	extreme_min_max_value minimal and maximal values	✔ OK
▶	positive_only shuffled sequence of 0...100 and then 102...200	✔ OK
▶	negative_only shuffled sequence -100 ... -1	✔ OK
expand all	**Performance tests**	
▶	medium chaotic sequences length=10005 (with minus)	✔ OK
▶	large_1 chaotic + sequence 1, 2, ..., 40000 (without minus)	✔ OK
▶	large_2 shuffled sequence 1, 2, ..., 100000 (without minus)	✔ OK
▶	large_3 chaotic + many -1, 1, 2, 3 (with minus)	✔ OK