

Candidate Report: Anonymous

[Check out Codility training tasks](#)

Test Name:

Summary

Timeline

Tasks summary

Task	Time spent	Score
MissingInteger Java 8	2 min	100%

Total score



Tasks Details

Medium	1. <a href="#">MissingInteger</a>	Task Score	Correctness	Performance
	Find the smallest positive integer that does not occur in a given sequence.	100%	100%	100%

Task description

Solution

This is a demo task.

Write a function:

```
class Solution { public int solution(int[] 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.

Programming language used: Java 8

Total time used: 2 minutes



Effective time used: 2 minutes



Notes: *not defined yet*

Task timeline



12:50:08

12:51:47

Code: 12:51:47 UTC, java, final, score:  
100

[show code in pop-up](#)

```
1  import java.util.*;
2  class Solution {
3      public int solution(int[] A) {
4          List<Integer> targetList = new ArrayList<>(A.length);
5          for (int i : A) {
6              targetList.add(i);
7          }
8          Collections.sort(targetList);
9          if ((targetList.get(targetList.size() - 1) < 1) ||
10             (targetList.get(0) > 1)) {
11              return 1;
12          }
13          Iterator<Integer> it = targetList.iterator();
14          int first = it.next();
15          int previousPositive = first > 0 ? first : 0;
16          while (it.hasNext()) {
17              int current = it.next();
18              if (current > 0) {
19                  if ((current > 1) && (previousPositive < 1)) {
20                      return 1;
21                  }
22                  if (current - previousPositive > 1) {
23                      return previousPositive + 1;
```

```

24         }
25         previousPositive = current;
26     }
27 }
28 return previousPositive + 1;
29 }
30 }

```

## 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	✓ OK
	chaotic sequences length=10005 (with minus)	
▶	large_1	✓ OK
	chaotic + sequence 1, 2, ..., 40000 (without minus)	
▶	large_2	✓ OK
	shuffled sequence 1, 2, ..., 100000 (without minus)	
▶	large_3	✓ OK
	chaotic + many -1, 1, 2, 3 (with minus)	

---

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.