codility

Check out Codility training tasks

Candidate Report: Anonymous

Test Name:

Summary Timeline

Test Score

50 out of 100 points

50%

Tasks in Test

	Time Spent	Task Score
PermMissingElem Submitted in: Java 8	21 min	50%

TASKS DETAILS

EASY

1. PermMissingElem
Find the missing element in a given permutation.

Task Score	Correctness	Performance
50%	20%	80%

Task description

An array A consisting of N different integers is given. The array contains integers in the range [1..(N + 1)], which means that exactly one element is missing.

Your goal is to find that missing element.

Write a function:

```
class Solution { public int solution(int[] A); }
```

that, given an array A, returns the value of the missing element.

For example, given array A such that:

```
A[0] = 2
A[1] = 3
A[2] = 1
A[3] = 5
```

the function should return 4, as it is the missing element.

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

- N is an integer within the range [0..100,000];
- the elements of A are all distinct;
- each element of array A is an integer within the range [1..(N + 1)].

Solution

Programming language used:	Java 8
Total time used:	21 minutes
Effective time used:	21 minutes
Notes:	*not defined yet*

Task timeline

09:21:42 09:41:54

Code: 09:41:53 UTC, java, final, score: **50**

show code in pop-up

```
1  // you can also use imports, for example:
2  // import java.util.*;
3
4  // you can write to stdout for debugging purposes, e.g.
5  // System.out.println("this is a debug message");
6
7  class Solution {
8      public static int max = 1000000;
9          public static boolean[] mem = new boolean[max + 1]
```

```
10
11          public static int solution(int[] a) {
12                  if (a.length == 0)
13                          return 1;
14
15                  for (int i = 0; i < a.length; i++) {
16                          mem[a[i]] = true;
17                  }
18
19                  //System.out.println(Arrays.toString(mem))
20
21                  for (int i = 1; i < a.length + 1; i++) {
22                          if (!mem[i])
23                                  return i;
24                  }
25
26                  return -1;
27          }
28  }
```

Analysis summary

The following issues have been detected: wrong answers.

For example, for the input `[1]` the solution returned a wrong answer (got -1 expected 2).

Analysis

expand all	Example tests	
▶ example example test	✓ OK	
expand all	Correctness tests	
▶ empty_and_single empty list and single element	✗ WRONG ANSWER	got -1 expected 2
▶ missing_first_or_last the first or the last element is missing	✗ WRONG ANSWER	got -1 expected 6
▶ single single element	✗ WRONG ANSWER	got -1 expected 2
▶ double two elements	✗ WRONG ANSWER	got -1 expected 3
▶ simple simple test	✓ OK	
expand all	Performance tests	
▶ medium1 medium test, length = ~10,000	✓ OK	
▶ medium2 medium test, length = ~10,000	✓ OK	
▶ large_range range sequence, length = ~100,000	✗ WRONG ANSWER	got -1 expected 100001
▶ large1 large test, length = ~100,000	✓ OK	
▶ large2 large test, length = ~100,000	✓ OK	