codility

Training center

Check out Codility training tasks

TASKS DETAILS

EASY

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

Task Score	Correctness	Performance
100%	100%	100%

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:

```
def solution(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.

Assume that:

- 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)].

Complexity:

- expected worst-case time complexity is O(N);
- expected worst-case space complexity is O(1) (not counting the storage required for input arguments).

Solution

Programming language used:	Python
Total time used:	2 minutes
Effective time used:	2 minutes
Notes:	*not defined yet*

Task timeline

04:54:12 04:55:46

Code: 04:55:46 UTC, py, final, score: **100** show code in pop-up

```
1  def solution(A):
2      total = len(A)
3      items_sorted = sorted(A)
4      for position in range(1, total+1):
5          if items_sorted[position-1] == positior
6              continue
7          else:
8              return position
9      return total + 1
```

Analysis summary

The solution obtained perfect score.

Analysis

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

expand all	**Example tests**	
▶	example example test	✔ OK
expand all	**Correctness tests**	
▶	empty_and_single empty list and single element	✔ OK
▶	missing_first_or_last the first or the last element is missing	✔ OK
▶	single single element	✔ OK
▶	double two elements	✔ OK
▶	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	✔ OK
▶	large1 large test, length = ~100,000	✔ OK
▶	large2 large test, length = ~100,000	✔ OK