codility

Check out Codility training tasks

Candidate Report: Anonymous

Test Name:

Summary Timeline

Test Score

100 out of 100 points

100%

Tasks in Test

	Time Spent	Task Score
PermCheck Submitted in: Python	5 min	100%

TASKS DETAILS

EASY	Task Score	Correctness	Performance
1. PermCheck Check whether array A is a permutation.	100%	100%	100%

Task description

A non-empty array A consisting of N integers is given.

A *permutation* is a sequence containing each element from 1 to N once, and only once.

For example, array A such that:

```
A[0] = 4
A[1] = 1
A[2] = 3
A[3] = 2
```

is a permutation, but array A such that:

```
A[0] = 4
A[1] = 1
A[2] = 3
```

is not a permutation, because value 2 is missing.

The goal is to check whether array A is a permutation.

Write a function:

```
def solution(A)
```

that, given an array A, returns 1 if array A is a permutation and 0 if it is not.

For example, given array A such that:

```
A[0] = 4
A[1] = 1
A[2] = 3
A[3] = 2
```

the function should return 1.

Solution

Programming language used:	Python
Total time used:	5 minutes
Effective time used:	5 minutes
Notes:	*not defined yet*

Task timeline

12:10:56 12:15:34

Code: 12:15:33 UTC, py, final, score: **100**

show code in pop-up

```
def solution(a):
    max_elem = max(a)
    if max_elem != len(a):
        return 0
    data_set = set(a)
    complete_set = set(range(1, max_elem + 1))
    return 1 if data_set == complete_set else 0
```

Given array A such that:

```
    A[0] = 4
    A[1] = 1
    A[2] = 3
```

the function should return 0.

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..1,000,000,000].

Analysis summary

The solution obtained perfect score.

Analysis

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

expand all	Example tests	
▶	example1 the first example test	✓ OK
▶	example2 the second example test	✓ OK
expand all	Correctness tests	
▶	extreme_min_max single element with minimal/maximal value	✓ OK
▶	single single element	✓ OK
▶	double two elements	✓ OK
▶	antiSum1 total sum is correct, but it is not a permutation, N <= 10	✓ OK
▶	small_permutation permutation + one element occurs twice, N = ~100	✓ OK
▶	permutations_of_ranges permutations of sets like [2..100] for which the anwsers should be false	✓ OK
expand all	Performance tests	
▶	medium_permutation permutation + few elements occur twice, N = ~10,000	✓ OK
▶	antiSum2 total sum is correct, but it is not a permutation, N = ~100,000	✓ OK
▶	large_not_permutation permutation + one element occurs three times, N = ~100,000	✓ OK
▶	large_range sequence 1, 2, ..., N, N = ~100,000	✓ OK
▶	extreme_values all the same values, N = ~100,000	✓ OK
▶	various_permutations all sequences are permutations	✓ OK

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.