Codility_

CodeCheck Report: trainingN96ZY2-TZN

Test Name:

Check out Codility training tasks

Summary | Timeline

Tasks summary

Task	Time spent	Score
PermCheck Python ⚠	6 min	100%

Total score

100%

Tasks Details

Easy

1. PermCheck
Check whether array A is a permutation.

Task Score	Correctness	Performance
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.

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].

Solution

Programming language used:	Python	
Total time used:	6 minutes	?
Effective time used:	6 minutes	?
Notes:	*not defined yet*	

Task timeline

18:56:48 19:02:38

Code: 19:02:38 UTC, py, final, score: 100 show code in pop-up

```python
# you can write to stdout for debugging purposes, e.g.
# print("this is a debug message")

def solution(A):
    # write your code in Python 3.6
    if len(A) == 1:
        if A[0] == 1:
            return 1
        else:
            return 0
    elif max(A) > len(A):
        return 0
    else:
        set_ = set(A)
        B = [i+1 for i in range(len(A))]
        set1 = set(B)
        if set_ == set1:
            return 1
        else:
            return 0
```

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