

## CodeCheck Report: training9NSM5J-4Q8

Test Name:

[Check out Codility training tasks](#)

Summary

Timeline

## Tasks summary

Task	Time spent	Score
PermCheck Python	1 min	100%

## Total score

100%

## Tasks Details

Easy	1. <b>PermCheck</b>	Task Score	Correctness	Performance	
	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
```

## Solution

Programming language used: Python

Total time used: 1 minutes ?

Effective time used: 1 minutes ?

Notes: not defined yet

## Task timeline

?



$A[2] = 3$

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].

Copyright 2009–2023 by Codility Limited. All Rights Reserved.

Unauthorized copying, publication or disclosure prohibited.

12:05:05

12:05:34

Code: 12:05:33 UTC, py, [show code in pop-up](#)  
final, score: 100

```
1 # you can write to stdout for debugging
2 # print("this is a debug message")
3
4 def solution(A):
5     # Implement your solution here
6     # pass
7     length_of_array = len(A)
8     sum_of_array = length_of_array * (1
9     if sum(A) == sum_of_array and len(s
10         return 1
11     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	✓ OK
the first example test	
▶ example2	✓ OK
the second example test	
expand all	Correctness tests
▶ extreme_min_max	✓ OK
single element with minimal/maximal value	
▶ single	✓ OK
single element	
▶ double	✓ OK
two elements	
▶ antiSum1	✓ OK
total sum is correct, but it is not a permutation, $N \leq 10$	
▶ small_permutation	✓ OK
permutation + one element occurs twice, $N \sim 100$	
▶ permutations_of_ranges	✓ OK

permutations of sets like [2..100]  
for which the answers should be  
false

expand all

#### Performance tests

▶	medium_permutation	✓ OK
	permutation + few elements occur twice, N = ~10,000	
▶	antiSum2	✓ OK
	total sum is correct, but it is not a permutation, N = ~100,000	
▶	large_not_permutation	✓ OK
	permutation + one element occurs three times, N = ~100,000	
▶	large_range	✓ OK
	sequence 1, 2, ..., N, N = ~100,000	
▶	extreme_values	✓ OK
	all the same values, N = ~100,000	
▶	various_permutations	✓ OK
	all sequences are permutations	