Codility_

CodeCheck Report: trainingD7Y3U3-3ER

Test Name:

Check out Codility training tasks

Summary Timeline AI Assistant Transcript

Tasks summary

Task	Time spent	Score
PermMissingElem C++	2 min	100%

Total score

Tasks Details

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

Easy

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:

```
int solution(vector<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.

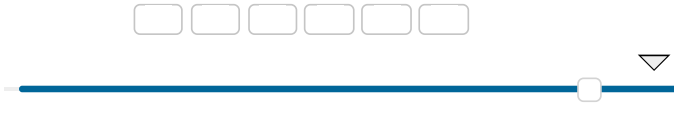
Write an **efficient** algorithm for the following assumptions:

- N is an integer within the range [0..100,000];
- the elements of A are all distinct;

Solution

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

Task timeline



13:01:13 13:03:04

Code: 13:03:03 UTC, cpp, final, score: **100** show code in pop-up

- each element of array A is an integer within the range [1..(N + 1)].


```
1  // you can use includes, for example:
2  // #include <algorithm>
3
4  // you can write to stdout for debugging purp
5  // cout << "this is a debug message" << endl;
6
7  int solution(vector<int> &A) {
8      int N = A.size();
9      long long sum = (long long)(N + 1) * (N +
10
11     for (int e : A) {
12         sum -= e;
13     }
14
15     return (int)sum;
16 }
```

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