Codility_

CodeCheck Report: trainingTF3VHE-DKF

Test Name:

Check out Codility training tasks

Summary Timeline

Tasks summary

Task	Time spent	Score
PassingCars Java 8 ⚠	19 min	100%

Total score

100%

Tasks Details

Easy

1. PassingCars
Count the number of passing cars on the road.

Task Score	Correctness	Performance
100%	100%	100%

Task description

A non-empty array A consisting of N integers is given. The consecutive elements of array A represent consecutive cars on a road.

Array A contains only 0s and/or 1s:

- 0 represents a car traveling east,
- 1 represents a car traveling west.

The goal is to count passing cars. We say that a pair of cars (P, Q), where 0 ≤ P < Q < N, is passing when P is traveling to the east and Q is traveling to the west.

For example, consider array A such that:

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

We have five pairs of pa

Write a function:

```
class Solution {
```

that, given a non-empty

of pairs of passing cars

Solution

Programming language used:	Java 8	
Total time used:	19 minutes	?
Effective time used:	19 minutes	?
Notes:	*not defined yet*	

Task timeline ?

17:54:12 18:13:07

show code in pop-up

```
or example:

 debugging purposes,
 a debug message");
```

How likely are you to recommend Codility to your friends and colleagues? ×

0 1 2 3 4 5 6 7 8 9 10

Not at all likely *Extremely likely*

The function should return −1 if the number of pairs of passing cars exceeds 1,000,000,000.

For example, given:

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

the function should return 5, as explained above.

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 that can have one of the following values: 0, 1.

```

class Solution {
    public int solution(int[] A) {
        final int MAX_RESULT = 1000000000;
        int result = 0, one_counter=0;
        for(int i=A.length-1; i>=0; i--) {
            if(A[i]==1) {
                one_counter++;
            }
            else {
                if(result>MAX_RESULT) {
                    return -1;
                }
                result+=one_counter;
            }
        }
        return result;
    }
}
```

Analysis summary

The solution obtained perfect score.

Analysis

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

expand all	Example tests
▶ example example test	✓ OK
expand all	Correctness tests
▶ single single element	✓ OK
▶ double two elements	✓ OK
▶ simple simple test	✓ OK
▶ small_random random, length = 100	✓ OK
▶ small_random2 random, length = 1000	✓ OK
expand all	Performance tests
▶ medium_random random, length = ~10,000	✓ OK
▶ large_random random, length = ~100,000	✓ OK
▶ large_big_answer 0..01..1, length = ~100,000	✓ OK
▶ large_alternate 0101..01, length = ~100,000	✓ OK
▶ large_extreme	✓ OK

How likely are you to recommend Codility to your friends and colleagues? ×

0 1 2 3 4 5 6 7 8 9 10

Not at all likely *Extremely likely*