Codility_

Candidate Report: trainingK6GP9H-QUJ

Check out Codility training tasks

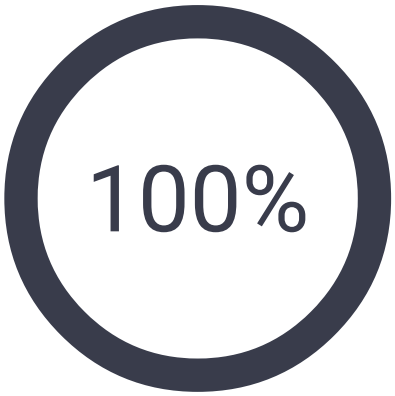
Test Name:

Summary Timeline

Tasks summary

Task	Time spent	Score
PassingCars Python	29 min	100%

Total score



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
```

Solution

Programming language used:	Python
Total time used:	29 minutes
Effective time used:	29 minutes
Notes:	*not defined yet*

Task timeline

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

We have five pairs of passing cars: (0, 1), (0, 3), (0, 4), (2, 3), (2, 4).

Write a function:

```
def solution(A)
```

that, given a non-empty array A of N integers, returns the number of pairs of passing cars.

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.

02:39:39 03:07:43

Code: 03:07:43 UTC, py, final, score: **100** show code in pop-up

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

def prefix_sums(A):
    P = [0]
    for i in range(len(A)):
        P += [P[i] + A[i]]
    return P

def solution(A):
    # write your code in Python 3.6
    ret = 0
    PreSum = prefix_sums(A)
    TotalSum = PreSum[len(A)]
    for i in range(len(A)):
        if 0 == A[i]:
            ret += TotalSum - PreSum[i]
            if ret > 1000000000:
                return -1
    return ret
```

Analysis summary

The solution obtained perfect score.

Analysis

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

collapse all — Example tests

▼ example — ✓ OK
example test

1. 0.036 s OK

collapse all — Correctness tests

▼ single — ✓ OK
single element

1. 0.036 s OK
2. 0.036 s OK

▼ double — ✓ OK
two elements

1. 0.036 s OK
2. 0.036 s OK
3. 0.036 s OK

4. 0.044 s **OK**

▼ simple ✓ OK
simple test

1. 0.036 s **OK**

▼ small_random ✓ OK
random, length = 100

1. 0.036 s **OK**

▼ small_random2 ✓ OK
random, length = 1000

1. 0.036 s **OK**

collapse all Performance tests

▼ medium_random ✓ OK
random, length = ~10,000

1. 0.048 s **OK**

▼ large_random ✓ OK
random, length = ~100,000

1. 0.148 s **OK**

▼ large_big_answer ✓ OK
0..01..1, length = ~100,000

1. 0.156 s **OK**
2. 0.120 s **OK**

▼ large_alternate ✓ OK
0101..01, length = ~100,000

1. 0.156 s **OK**
2. 0.144 s **OK**

▼ large_extreme ✓ OK
large test with all 1s/0s, length = ~100,000

1. 0.168 s **OK**
2. 0.156 s **OK**
3. 0.164 s **OK**
