codility

Training center

dility training tasks

Congratulations

You have completed a Codility demo.

Tweet this!
I scored 100% in #ruby on @Codility!
https://codility.com/demo/take-sample-test/passing_cars/

Sign up for our newsletter!

Like us on Facebook!

Demo ticket

Session

ID: demoBQAT8V-36J
Time limit: 120 min.

Status: closed

Created on: 2014-12-20 12:46 UTC
Started on: 2014-12-20 12:48 UTC
Finished on: 2014-12-20 14:06 UTC

Tasks in test	Correctness	Performance	Task score
1 | PassingCars	100%	100%	100%

Test score

100%

100 out of 100 points

EASY

1. PassingCars

Count the number of passing cars on the road.

score: 100 of 100

Task description

A non-empty zero-indexed 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 passing cars: (0, 1), (0, 3), (0, 4), (2, 3), (2, 4).
Write a function:

```
def solution(a)
```

that, given a non-empty zero-indexed array A of N integers, returns the number of passing cars.
The function should return −1 if the number 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.
Assume that:

- 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.

Solution

Programming language used: Ruby

Total time used: 79 minutes

Effective time used: 79 minutes

Notes: *not defined yet*

Task timeline

Code: 14:06:19 UTC, rb, final, score: **100.00**

```ruby
# you can use puts for debugging purposes, e.g.
# puts "this is a debug message"

def solution(a)
  total_east = 0
  total = 0
  a.each_with_index do |car, index|
    if car == 0
      total_east += 1
    else
      total += total_east
      return -1 if total > 1_000_000_000
    end
  end
  total
end
```

Analysis

Complexity:

- expected worst-case time complexity is O(N);
- expected worst-case space complexity is O(1), beyond input storage (not counting the storage required for input arguments).

Elements of input arrays can be modified.

Detected time complexity:

O(N)

test	time	result
Example tests		
example example test	0.064 s	**OK**
Correctness tests		
single single element	0.064 s	**OK**
double two elements	0.060 s	**OK**
simple simple test	0.056 s	**OK**
small_random random, length = 100	0.056 s	**OK**
Performance tests		
medium_random random, length = ~10,000	0.076 s	**OK**
large_random random, length = ~100,000	0.200 s	**OK**
large_big_answer 0..01..1, length = ~100,000	0.160 s	**OK**
large_alternate 0101..01, length = ~100,000	0.192 s	**OK**
large_extreme large test with all 1s/0s, length = ~100,000	0.208 s	**OK**

Training center