Demo ticket

Session

ID: demoNBEZ9P-43M
Time limit: 120 min.

Status: closed

Created on: 2014-03-16 22:33 UTC
Started on: 2014-03-16 22:33 UTC
Finished on: 2014-03-16 23:06 UTC

Tasks in test

Task score

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:

```
    int solution(vector<int> &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];

Solution

Programming language used: C++

Total time used: 33 minutes ?

Effective time used: 33 minutes ?

Notes: correct functionality and scalability

Task timeline ?

22:33:46 23:06:36

Code: 23:06:36 UTC, cpp, final, score: **100.00**

```
// you can also use includes, for example:
// #include <algorithm>
int solution(vector<int> &A) {
    // write your code in C++98
    long int npass = 0;
    long int nw = 0;
    long int ne = 0;

    for (int i = 0; i < (int)A.size(); i++) {
        // east
        if (A[i] == 0) {
            ++ne;
        // west
        } else if (A[i] == 1) {
            ++nw;
            npass += ne;
            if (npass > 1000000000)
                return -1;
        // invalid
        } else
```

- each element of array A is an integer within the range [0..1].

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.

```
            return -1;
    }
    return npass;
}
```

Analysis

Detected time complexity:

O(N)

test	time	result
example example test	0.020 s.	**OK**
single single element	0.020 s.	**OK**
double two elements	0.020 s.	**OK**
simple simple test	0.020 s.	**OK**
small_random random, length = 100	0.020 s.	**OK**
medium_random random, length = ~10,000	0.020 s.	**OK**
large_random random, length = ~100,000	0.040 s.	**OK**
large_big_answer 0..01..1, length = ~100,000	0.030 s.	**OK**
large_alternate 0101..01, length = ~100,000	0.030 s.	**OK**
large_extreme large test with all 1s/0s, length = ~100,000	0.030 s.	**OK**

Training center