Codility_

CodeCheck Report: trainingMACYEQ-36R

Test Name:

Summary   Timeline

Tasks summary

Task	Time spent	Score
Fish C# ⚠	24 min	100%

Total score

100%

Tasks Details

1. Fish

Easy

N voracious fish are moving along a river. Calculate how many fish are alive.

Task Score	Correctness	Performance
100%	100%	100%

Task description

You are given two non-empty arrays A and B consisting of N integers. Arrays A and B represent N voracious fish in a river, ordered downstream along the flow of the river.

The fish are numbered from 0 to N − 1. If P and Q are two fish and P < Q, then fish P is initially upstream of fish Q. Initially, each fish has a unique position.

Fish number P is represented by A[P] and B[P]. Array A contains the sizes of the fish. All its elements are unique. Array B contains the directions of the fish. It contains only 0s and/or 1s, where:

- 0 represents a fish flowing upstream,
- 1 represents a fish flowing downstream.

If two fish move in opposite directions and there are no other (living) fish between them, they will eventually meet each other. Then only one fish can s... one. More precisely, we... other when P < Q, B[P] =... between them. After the...

- If A[P] > A[Q] the... flowing downstr...
- If A[Q] > A[P] the... flowing upstrea...

Solution

Programming language used:	C#
Total time used:	24 minutes ❓
Effective time used:	24 minutes ❓
Notes:	*not defined yet*

Task timeline ❓
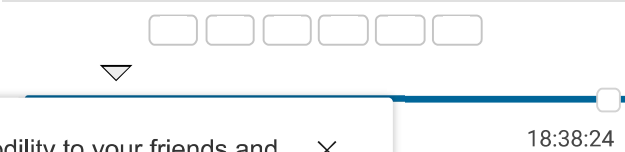
18:38:24

show code in pop-up

...rts, for example: ...eneric;

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*

We assume that all the fish are flowing at the same speed. That is, fish moving in the same direction never meet. The goal is to calculate the number of fish that will stay alive.

For example, consider arrays A and B such that:

```
A[0] = 4     B[0] = 0
A[1] = 3     B[1] = 1
A[2] = 2     B[2] = 0
A[3] = 1     B[3] = 0
A[4] = 5     B[4] = 0
```

Initially all the fish are alive and all except fish number 1 are moving upstream. Fish number 1 meets fish number 2 and eats it, then it meets fish number 3 and eats it too. Finally, it meets fish number 4 and is eaten by it. The remaining two fish, number 0 and 4, never meet and therefore stay alive.

Write a function:

```
class Solution { public int solution(int[] A,
int[] B); }
```

that, given two non-empty arrays A and B consisting of N integers, returns the number of fish that will stay alive.

For example, given the arrays shown above, the function should return 2, 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 within the range [0..1,000,000,000];
- each element of array B is an integer that can have one of the following values: 0, 1;
- the elements of A are all distinct.

```
4
5     // you can write to stdout for debugging purposes,
6     // Console.WriteLine("this is a debug message");
7
8
9     using System;
10    using System.Collections.Generic;
11
12    class Solution {
13        public int solution(int[] A, int[] B) {
14            int N = A.Length;
15
16            Stack<int> upStream = new Stack<int>();
17            Stack<int> downStream = new Stack<int>();
18
19            for (int i = 0; i < N; i++) {
20                if ((B[i] == 0)) {
21                    upStream.Push(A[i]);
22
23                    while (downStream.Count > 0) {
24                        if (downStream.Peek() < A[i]) {
25                            downStream.Pop();
26                        } else {
27                            upStream.Pop();
28                            break;
29                        }
30                    }
31                } else {
32                    downStream.Push(A[i]);
33                }
34            }
35
36            return downStream.Count + upStream.Count;
37        }
38    }
39
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: $O(N)$

expand all	Example tests	
▶ example		✓ OK
example test		
expand all	Correctness tests	
▶ extreme_small		✓ OK
1 or 2 fishes		
▶ simple1		✓ OK
simple test		
▶ simple2		✓ OK
simple test		
▶ small_random		✓ 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*

direction

▶ extreme_range2 ✓ OK

all fish flowing in the same direction

Not at all likely

Extremely likely

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