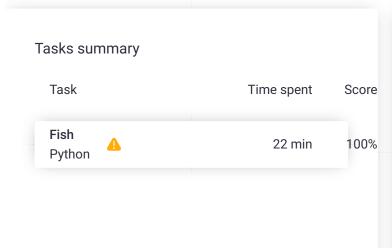
Codility_

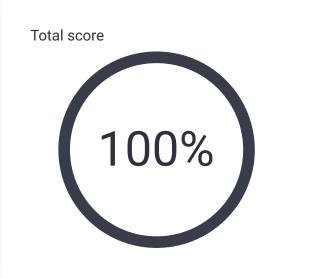
CodeCheck Report: trainingMWU9YF-FDR

Test Name:

Check out Codility training tasks

Summary Timeline





Tasks Details

1. Fish

Ν

voracious fish are

moving along a

river.
Calculate
how
many fish
are alive.

Task Score

Correctness

100%

Performance

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.

Solution

Programming language used: Python

Total time used: 22 minutes 2

Effective time used: 22 minutes 3

1 von 3

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 stay alive – the larger fish eats the smaller one. More precisely, we say that two fish P and Q meet each other when P < Q, B[P] = 1 and B[Q] = 0, and there are no living fish between them. After they meet:

- If A[P] > A[Q] then P eats Q, and P will still be flowing downstream,
- If A[Q] > A[P] then Q eats P, and Q will still be flowing upstream.

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:

```
def solution(A, 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];

```
not defined yet
 Notes:
Task timeline
 08:28:49
                                      08:50:42
  Code: 08:50:41 UTC,
                          show code in pop-up
  py, final, score: 100
      # you can write to stdout for debuggi
  1
  2
      # print("this is a debug message")
  3
  4
      def solution(A, B):
  5
           # Implement your solution here
  6
           # pass
  7
           stack = []
  8
           remaining_fish = 0
  9
 10
           for i in range(len(A)):
 11
               if B[i] == 0:
 12
                   while stack and A[i] > st
 13
                        stack.pop()
 14
 15
                   if not stack:
 16
                        remaining_fish += 1
 17
               else:
 18
                   stack.append(A[i])
 19
 20
           return remaining_fish + len(stack
```

Analysis summary

The solution obtained perfect score.

Analysis

Detected time complexity: **O**



expand all	Examp	le tests
example		V
example test		OK
expand all	Correctr	ess tests
extreme_sr	mall	V
1 or 2 fishes		ок
► simple1		~
simple test		ок
▶ simple2		✓

2 von 3 18.07.23, 10:52

- 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.

Copyright 2009–2023 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

simp	ole test	ок
>	small_random small random test, N = ~100	ok
expa	and all Performan	ce tests
>	medium_random small medium test, N = ~5,0	∨ 00 OK
>	large_random large random test, N = ~100,	000 OK
>	extreme_range1 all except one fish flowing in same direction	the OK
>	extreme_range2 all fish flowing in the same direction	ok

3 von 3