

## Demo ticket

### Session

ID: demo8QMXH2-E7M  
Time limit: 120 min.

### Status: closed

Created on: 2014-03-20 02:39 UTC  
Started on: 2014-03-20 03:28 UTC  
Finished on: 2014-03-20 03:41 UTC

### Tasks in test

1 | Fish

### Task score

100%

### Test score

100%  
100 out of 100 points

MEDIUM

### 1. Fish

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

score: 100 of 100



#### Task description

You are given two non-empty zero-indexed 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, fish number P is represented by A[P] and B[P], and if P < Q then fish P is initially upstream of fish Q. Initially, each fish has a unique position.

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, numbers 0 and 4, never

#### Solution

Programming language used: Python

Total time used: 14 minutes

Effective time used: 14 minutes

Notes: correct functionality and scalability

#### Task timeline



03:28:25

03:41:42

Code: 03:41:42 UTC, py, final, score: 100.00

```
01. def solution(A, B):
02.     N = len(A)
03.     count = 0
04.     stack = []
05.     for i in xrange(N):
06.         if B[i] == 0:
07.             if not stack:
08.                 count += 1
09.             else:
10.                 while stack:
11.                     if stack[-1] < A[i]:
12.                         stack.pop()
13.                     else:
14.                         break
15.                 if not stack:
16.                     count += 1
17.             else:
18.                 stack.append(A[i])
19.
20.     return len(stack) + count
21.
22.
```

meet and therefore stay alive.  
Write a function:

```
def solution(A, B)
```

that, given two non-empty zero-indexed 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.  
Assume that:

- 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.

Complexity:

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

Elements of input arrays can be modified.

Copyright 2009–2014 by Codility Limited. All Rights Reserved. Unauthorized copying, publication or disclosure prohibited.

Analysis



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

test	time	result
example example test	0.050 s.	OK
extreme_small 1 or 2 fishes	0.050 s.	OK
simple1 simple test	0.050 s.	OK
simple2 simple test	0.050 s.	OK
small_random small random test, N = ~100	0.050 s.	OK
medium_random small medium test, N = ~5,000	0.070 s.	OK
large_random large random test, N = ~100,000	0.410 s.	OK
extreme_range1 all except one fish flowing in the same direction	0.370 s.	OK
extreme_range2 all fish flowing in the same direction	0.400 s.	OK

Training center