AVAILABLE LESSONS:

Lesson 1
Iterations

Lesson 2
Arrays

Lesson 3
Time Complexity

Lesson 4
Counting Elements

Lesson 5
Prefix Sums

Lesson 6
Sorting

Lesson 7
Stacks and Queues

Lesson 8
Leader

Lesson 9
Maximum slice problem

Lesson 10
Prime and composite numbers

Lesson 11

PAINLESS

Fish

START

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

Programming language: C++

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. 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 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
```

Sieve of Eratosthenes

Lesson 12
Euclidean algorithm

Lesson 13
Fibonacci numbers

Lesson 14
Binary search algorithm

Lesson 15
Caterpillar method

Lesson 16
Greedy algorithms

Lesson 17
Dynamic programming

Lesson 90
Tasks from Indeed Prime 2016 challenge

Lesson 99
Future training

```
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:

```
int solution(vector<int> &A, vector<int> &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.

For programmers

Lessons Challenges
Terms FAQ

For companies

Tour About us
Pricing Jobs
Blog Terms
Privacy Cookies
API

Sign up for our newsletter:

Information about upcoming challenges, solutions and lessons directly in your inbox.

Your email Sign up

Social:

Contact us:

For customer support queries:
UK +44 (0) 208 970 78 68
US 1-415-466-8085
support@codility.com

For sales queries:
UK +44 (0) 208 970 78 67
US 1-415-466-8085
sales@codility.com