Offline Coding Challenges

2 exercises, 45 minutes each.

# Road Pavement

The mayor of a city is paving a road and wants to know when the work will be done. The road has a length of X kilometers.

A zero indexed array A consisting of N integers representing the months when the contractor can have a kilometer of the road paved is given.

The goal is to find the earliest month when the road gets paved completely.

For example, you are given integer X = 5 and array A such that:

A[0] = 5 // in month 0 the contractor can have kilometer 5 paved

A[1] = 3 // in month 1 the contractor can have kilometer 3 paved

A[2] = 4 // in month 2 the contractor can have kilometer 4 paved

A[3] = 4 // in month 3 the contractor can have kilometer 4 paved

A[4] = 2 // in month 4 the contractor can have kilometer 2 paved

A[5] = 3 // in month 5 the contractor can have kilometer 3 paved

A[6] = 1 // in month 6 the contractor can have kilometer 1 paved

A[7] = 4 // in month 7 the contractor can have kilometer 4 paved

In month 6, the contractor can finish paving kilometer 1 of the road. This is the earliest month when pavement is finished in every kilometer across the road.

Write a function in C#:

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

that, given a non-empty zero-indexed array A consisting of N integers and integer X, returns the earliest time when the contractor finishes paving the road.

If the contractor will never be able to completely pave the road, the function should return −1.

For example, given X = 5 and array A such that:

A[0] = 5

A[1] = 3

A[2] = 4

A[3] = 4

A[4] = 2

A[5] = 3

A[6] = 1

A[7] = 4

the function should return 6, as explained above.

Assume that:

N and X are integers within the range [1..100,000];

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

Complexity:

expected worst-case time complexity is O(N);

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

Elements of input arrays can be modified.

# Crossing Planes

A non-empty zero-indexed array A consisting of N integers is given. The consecutive elements of array A represent planes either going from New York to London or from London to New York. The index of the array represents the position along the route, 0 is almost New York and N-1 is almost London.

Array A contains only 0s and/or 1s:

* 0 represents a plane traveling from New York to London,
* 1 represents a plane traveling from London to New York.

Control towers on both airports want to know how many planes crossings will happen. We say that a pair of planes (P, Q), where 0 ≤ P < Q < N, are crossing each other when P is traveling from New York to London and Q is traveling from London to New York.

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 planes crossing: (0, 1), (0, 3), (0, 4), (2, 3), (2, 4).

Write a function in C#:

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

that, given a non-empty zero-indexed array A of N integers, returns the number of pairs of crossing planes.

The function should return −1 if the number of pairs of crossing planes 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];
* each element of array A is an integer that can have one of the following values: 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.