A non-empty zero-indexed array A consisting of N integers is given.

A *permutation* is a sequence containing each element from 1 to N once, and only once.

For example, array A such that:

A[0] = 4

A[1] = 1

A[2] = 3

A[3] = 2

is a permutation, but array A such that:

A[0] = 4

A[1] = 1

A[2] = 3

is not a permutation, because value 2 is missing.

The goal is to check whether array A is a permutation.

Write a function:

int solution(int A[], int N);

that, given a zero-indexed array A, returns 1 if array A is a permutation and 0 if it is not.

For example, given array A such that:

A[0] = 4

A[1] = 1

A[2] = 3

A[3] = 2

the function should return 1.

Given array A such that:

A[0] = 4

A[1] = 1

A[2] = 3

the function should return 0.

Assume that:

* N is an integer within the range [1..100,000];
* each element of array A is an integer within the range [1..1,000,000,000].

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.