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

The *leader* of this array is the value that occurs in more than half of the elements of A.

An *equi leader* is an index S such that 0 ≤ S < N − 1 and two sequences A[0], A[1], ..., A[S] and A[S + 1], A[S + 2], ..., A[N − 1] have leaders of the same value.

For example, given array A such that:

A[0] = 4

A[1] = 3

A[2] = 4

A[3] = 4

A[4] = 4

A[5] = 2

we can find two equi leaders:

* 0, because sequences: (4) and (3, 4, 4, 4, 2) have the same leader, whose value is 4.
* 2, because sequences: (4, 3, 4) and (4, 4, 2) have the same leader, whose value is 4.

The goal is to count the number of equi leaders.

Write a function:

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

that, given a non-empty zero-indexed array A consisting of N integers, returns the number of equi leaders.

For example, given:

A[0] = 4

A[1] = 3

A[2] = 4

A[3] = 4

A[4] = 4

A[5] = 2

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 [−1,000,000,000..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.