XOR Sharing

Alice and Bob are longtime friends that have recently become embroiled in a bitter dispute. They share an array of N bags of coins, with each bag holding some number of coins. So far unable to resolve their argument, they have taken each other to court.

The judge at court has finally made a ruling. He calls a contiguous subarray of bags **suitable** if there exist two non-empty disjoint subsets of bags (not necessarily using all the bags) that can be given to Alice and Bob, such that the bitwise XOR[1] of the coin counts in the bags for both subsets are equal. For example, if the counts of coins in a subarray of bags are $[1, 3, 7, 2, 2]$, then this is suitable because Alice can be given the 1st and 4th bags, while Bob can take the 2nd bag. The subarray $[1, 3, 5]$ is not suitable, since there is no way to find two disjoint subsets with equal XOR.

In the ruling, Alice and Bob will be given a suitable subarray of bags so that fairness is achieved. The judge is curious how many options of subarrays he has. Count the number of contiguous subarrays of the original array that are **suitable**.

Input

The first line of input contains a single integer N ($1 \leq N \leq 200\,000$). The second and final line of input contains N integers $a_1, a_2, \ldots, a_N$ ($0 \leq a_i \leq 10^9$). Each a_i is the number of coins in the i-th bag.

Output

Output a single integer representing the number of contiguous subarrays of the array that are suitable.

Sample Input 1

```
5
7 7 1 5 9
```

Sample Output 1

```
4
```

[1]The bitwise XOR of two numbers x and y is the unique number where each bit position is a 1 if and only if exactly one of the corresponding bits in x and y is also 1. This is the same as the $\hat{}$ operator in C++/Java/Python.