Problem I. We Need the Zero

Time limit 1000 ms
Mem limit 262144 kB

There is an array a consisting of non-negative integers. You can choose an integer x and denote $b_i = a_i \oplus x$ for all $1 \le i \le n$, where $\oplus$ denotes the bitwise XOR operation. Is it possible to choose such a number x that the value of the expression $b_1 \oplus b_2 \oplus \ldots \oplus b_n$ equals 0?

It can be shown that if a valid number x exists, then there also exists x such that ($0 \le x < 2^8$).

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 1000$). The description of the test cases follows.

The first line of the test case contains one integer n ($1 \le n \le 10^3$) — the length of the array a.

The second line of the test case contains n integers — array a ($0 \le a_i < 2^8$).

It is guaranteed that the sum of n over all test cases does not exceed 10^3.

Output

For each set test case, print the integer x ($0 \le x < 2^8$) if it exists, or -1 otherwise.

Sample 1

Input	Output

5
3
1 2 5
3
1 2 3
4
0 1 2 3
4
1 2 2 3
1
1
	6
	0
	3
	-1
	1

Note

In the first test case, after applying the operation with the number 6 the array b becomes $[7, 4, 3]$, $7 \oplus 4 \oplus 3 = 0$.

There are other answers in the third test case, such as the number 0.