B. Even Array

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given an array $a[0 \ldots n-1]$ of length n which consists of non-negative integers. **Note that array indices start from zero.**

An array is called *good* if the parity of each index matches the parity of the element at that index. More formally, an array is good if for all i $(0 \le i \le n-1)$ the equality $i \bmod 2 = a[i] \bmod 2$ holds, where $x \bmod 2$ is the remainder of dividing x by 2.

For example, the arrays $[0, 5, 2, 1]$ and $[0, 17, 0, 3]$ are good, and the array $[2, 4, 6, 7]$ is bad, because for $i = 1$, the parities of i and $a[i]$ are different: $i \bmod 2 = 1 \bmod 2 = 1$, but $a[i] \bmod 2 = 4 \bmod 2 = 0$.

In one move, you can take **any** two elements of the array and swap them (these elements are not necessarily adjacent).

Find the minimum number of moves in which you can make the array a good, or say that this is not possible.

Input

The first line contains a single integer t $(1 \le t \le 1000)$ — the number of test cases in the test. Then t test cases follow.

Each test case starts with a line containing an integer n $(1 \le n \le 40)$ — the length of the array a.

The next line contains n integers $a_0, a_1, \ldots, a_{n-1}$ $(0 \le a_i \le 1000)$ — the initial array.

Output

For each test case, output a single integer — the minimum number of moves to make the given array a good, or -1 if this is not possible.

Example

input
4 4 3 2 7 6 3 3 2 6 1 7 7 4 9 2 1 18 3 0

output
2 1 -1 0

Note

In the first test case, in the first move, you can swap the elements with indices 0 and 1, and in the second move, you can swap the elements with indices 2 and 3.

In the second test case, in the first move, you need to swap the elements with indices 0 and 1.

In the third test case, you cannot make the array good.