Non-adjacent Swaps

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

You are given a permutation. In one step, you can swap any two adjacent elements if their values differ by more than one.

Consider all permutations that can be obtained using these operations, starting from a given permutation. Define a *distance* between two permutations as the minimum number of these operations required to transform the first permutation into the second one.

Find the number of permutations that can be obtained and the sum of distances to all those permutations from the starting one.

Input

The first line contains the integer n ($1 \le n \le 100$) — the permutation size. The second line contains n distinct integers a_i ($1 \le a_i \le n$) — the starting permutation.

Output

Output two integers, the number of obtainable permutations, and the sum of distances to those permutations. Calculate both numbers modulo $1\,000\,000\,007$.

Scoring

Subtask	Points	Constraints
1	15	$n \le 8$
2	20	$n \le 15$
3	30	$n \le 30$
4	20	$n \le 50$
5	15	No additional constraints

Examples

standard input	standard output
4 3 1 4 2	5 5
5 1 2 3 4 5	1 0
6 5 3 1 2 4 6	61 218

Note

In the first example, you can obtain the following permutations:

- $[3, 1, 4, 2]$, in 0 steps;
- $[3, 1, 2, 4]$, in 1 step;
- $[1, 3, 4, 2]$, in 1 step;
- $[3, 4, 1, 2]$, in 1 step;

- $[1, 3, 2, 4]$, in 2 steps (swap two pairs of elements with values $(3, 1)$ and $(4, 2)$).

There are five permutations in total; the sum of distances also equals five.

In the second example, you can't perform any swap, so only one permutation is reachable with a distance of 0.