

C. Bear and Up-Down

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

The life goes up and down, just like nice sequences. Sequence t_1, t_2, \dots, t_n is called *nice* if the following two conditions are satisfied:

- $t_i < t_{i+1}$ for each odd $i < n$;
- $t_i > t_{i+1}$ for each even $i < n$.

For example, sequences (2, 8), (1, 5, 1) and (2, 5, 1, 100, 99, 120) are nice, while (1, 1), (1, 2, 3) and (2, 5, 3, 2) are not.

Bear Limak has a sequence of positive integers t_1, t_2, \dots, t_n . This sequence **is not nice** now and Limak wants to fix it by a single swap. He is going to choose two indices $i < j$ and swap elements t_i and t_j in order to get a nice sequence. Count the number of ways to do so. Two ways are considered different if indices of elements chosen for a swap are different.

Input

The first line of the input contains one integer n ($2 \leq n \leq 150\,000$) — the length of the sequence.

The second line contains n integers t_1, t_2, \dots, t_n ($1 \leq t_i \leq 150\,000$) — the initial sequence. It's guaranteed that the given sequence is not nice.

Output

Print the number of ways to swap two elements exactly once in order to get a nice sequence.

Examples

input
5 2 8 4 7 7
output
2
input
4 200 150 100 50
output
1
input
10 3 2 1 4 1 4 1 4 1 4
output
8
input
9 1 2 3 4 5 6 7 8 9

output
0

Note

In the first sample, there are two ways to get a nice sequence with one swap:

1. Swap $t_2 = 8$ with $t_4 = 7$.
2. Swap $t_1 = 2$ with $t_5 = 7$.

In the second sample, there is only one way — Limak should swap $t_1 = 200$ with $t_4 = 50$.