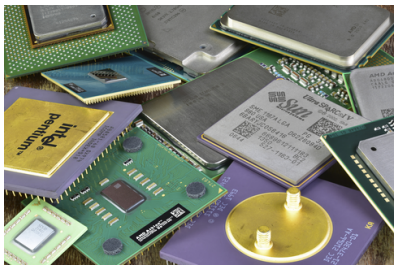
A Alternating Algorithm

Time limit: 8s



CPUs. CC0 by Martijn Boer on Flickr

In recent years, CPU manufacturers have found it increasingly difficult to keep up with Moore's law of doubling the number of transistors on integrated circuit chips every two years. To address this, manufacturers have instead started creating CPUs with an increasingly higher number of cores. In fact, you just purchased a CPU with a staggering n number of cores, no less!

Incidentally, you also have an array of $n + 1$ integers, $a_0, a_1, \ldots, a_n$, that you need to sort. To make good use of the large number of cores on your CPU, you have devised a parallel sorting algorithm in which there is a dedicated core for comparing each adjacent pair of integers. As long as the array is not sorted in non-decreasing order, the algorithm proceeds in rounds that alternate between:

- Odd rounds (starting with the first): The first core compares a_0 and a_1, the third core compares a_2 and a_3, the fifth core compares a_4 and a_5, and so on. If a pair of compared elements are out of order, the corresponding core will swap their positions. If n is even, a_n will be left untouched.

- Even rounds: The second core compares a_1 and a_2, the fourth core compares a_3 and a_4, the sixth core compares a_5 and a_6, and so on. If a pair of compared elements are out of order, the corresponding core will swap their positions. If n is odd, a_n will be left untouched, and a_0 will be left untouched no matter what the parity of n is.

Note that in both types of rounds some cores may be idle.

Before implementing this algorithm, you have decided to do some analysis. In particular, you noticed that the time complexity of the algorithm does not depend on the value of n, but rather it depends on the number of rounds that the algorithm runs. Given the initial contents of the array, determine the number of rounds that the parallel sorting algorithm runs before the array becomes sorted.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 4 \cdot 10^5$), the number of cores and the size of the array.

- One line with $n + 1$ integers $a_0, a_1, \ldots, a_n$ ($0 \leq a_i \leq 10^9$ for each i), the initial contents of the array.

Output

Output the number of rounds that the parallel sorting algorithm runs before the array becomes sorted in non-decreasing order.

Sample Input 1	Sample Output 1
3 8 13 4 10	3

Sample Input 2	Sample Output 2
5 13 12 14 10 14 12	3

Sample Input 3	Sample Output 3
2 2 2 1	3