

## Problem B: Ultra-QuickSort

In this problem, you have to analyze a particular sorting algorithm. The algorithm processes a sequence of *n* distinct integers by swapping two adjacent sequence elements until the sequence is sorted in ascending order. For the input sequence

9 1 0 5 4 ,

Ultra-QuickSort produces the output

0 1 4 5 9 .

Your task is to determine how many swap operations Ultra-QuickSort needs to perform in order to sort a given input sequence.

The input contains several test cases. Every test case begins with a line that contains a single integer *n < 500,000* -- the length of the input sequence. Each of the the following *n* lines contains a single integer *0 ≤ a[i] ≤ 999,999,999*, the *i*-th input sequence element. Input is terminated by a sequence of length *n = 0*. This sequence must not be processed.

For every input sequence, your program prints a single line containing an integer number *op*, the minimum number of swap operations necessary to sort the given input sequence.

### Sample Input

5

9

1

0

5

4

3

1

2

3

0

### Output for Sample Input

6

0

*Stefan B黷tcher*