Code Jam 2021 - Qualification Round

Reversort

Problem

Note: The main parts of the statements of the problems "Reversort" and "Reversort Engineering" are identical, except for the last paragraph. The problems can otherwise be solved independently.

Reversort is an algorithm to sort a list of distinct integers in increasing order. The algorithm is based on the "Reverse" operation. Each application of this operation reverses the order of some contiguous part of the list.

The pseudocode of the algorithm is the following:

```
Reversort(L):
  for i := 1 to length(L) - 1
    j := position with the minimum value in L between i and length(L), inclusive
    Reverse(L[i..j])
```

After $i-1$ iterations, the positions $1,\ 2,\ \ldots,\ i-1$ of the list contain the $i-1$ smallest elements of L, in increasing order. During the i-th iteration, the process reverses the sublist going from the i-th position to the current position of the i-th minimum element. That makes the i-th minimum element end up in the i-th position.

For example, for a list with 4 elements, the algorithm would perform 3 iterations. Here is how it would process $L = [4, 2, 1, 3]$:

1. $i = 1,\ j = 3 \longrightarrow L = [1, 2, 4, 3]$
2. $i = 2,\ j = 2 \longrightarrow L = [1, 2, 4, 3]$
3. $i = 3,\ j = 4 \longrightarrow L = [1, 2, 3, 4]$

The most expensive part of executing the algorithm on our architecture is the Reverse operation. Therefore, our measure for the cost of each iteration is simply the length of the sublist passed to Reverse, that is, the value $j - i + 1$. The cost of the whole algorithm is the sum of the costs of each iteration.

In the example above, the iterations cost 3, 1, and 2, in that order, for a total of 6.

Given the initial list, compute the cost of executing Reversort on it.

Input

The first line of the input gives the number of test cases, $\mathbf{T}$. $\mathbf{T}$ test cases follow. Each test case consists of 2 lines. The first line contains a single integer $\mathbf{N}$, representing the number of elements in the input list. The second line contains $\mathbf{N}$ distinct integers $\mathbf{L_1}$, $\mathbf{L_2}$, ..., $\mathbf{L_N}$, representing the elements of the input list L, in order.

Output

For each test case, output one line containing `Case #`x`:` y, where x is the test case number (starting from 1) and y is the total cost of executing Reversort on the list given as input.

Limits

Time limit: 10 seconds.
Memory limit: 1 GB.

Test Set 1 (Visible Verdict)

$1 \le \mathbf{T} \le 100$.
$2 \le \mathbf{N} \le 100$.
$1 \le \mathbf{L_i} \le N$, for all i.
$\mathbf{L_i} \ne \mathbf{L_j}$, for all $i \ne j$.

Sample

Sample Input

```
3
4
4 2 1 3
2
1 2
7
7 6 5 4 3 2 1
```

Sample Output

```
Case #1: 6
Case #2: 1
Case #3: 12
```

Sample Case #1 is described in the statement above.

In Sample Case #2, there is a single iteration, in which Reverse is applied to a sublist of size 1. Therefore, the total cost is 1.

In Sample Case #3, the first iteration reverses the full list, for a cost of 7. After that, the list is already sorted, but there are 5 more iterations, each of which contributes a cost of 1.