Advanced analysis of Insertion Sort

Insertion Sort is a simple sorting technique which was covered in previous challenges. Sometimes, arrays may be too large for us to wait around for insertion sort to finish. Is there some other way we can calculate the number of times Insertion Sort shifts each elements when sorting an array?

If k_i is the number of elements over which the i^{th} element of the array has to shift, then the total number of shifts will be $k_1 + k_2 + \ldots + k_N$.

Input Format

The first line contains a single integer, T, denoting the number of queries to perform. The $2T$ subsequent lines describe each query over two lines:

1. The first line contains an integer, N, denoting the number of elements to be sorted.
2. The second line contains N space-separated integers describing the respective values of $a[1], a[2], \ldots, a[N]$.

Constraints

$1 \leq T \leq 15$
$1 \leq N \leq 100000$
$1 \leq a[i] \leq 10000000$

Output Format

Print T lines containing the required answer for each query.

Sample Input 0

```
2
5
1 1 1 2 2
5
2 1 3 1 2
```

Sample Output 0

```
0
4
```

Explanation 0

The first query is already sorted, therefore there's no need to shift any element. In the second case, it will proceed in the following way.

```
Array: 2 1 3 1 2 -> 1 2 3 1 2 -> 1 1 2 3 2 -> 1 1 2 2 3
Moves: ---------- 1 ---------- 2 ---------- 1 --------- = 4
```