

Insertion Sort Advanced Analysis

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 shifts an insertion sort performs 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 $k[1] + k[2] + \dots + k[n]$.

Example

$arr = [4, 3, 2, 1]$

Array	Shifts
[4, 3, 2, 1]	
[3, 4, 2, 1]	1
[2, 3, 4, 1]	2
[1, 2, 3, 4]	3

Total shifts = 1 + 2 + 3 = 6

Function description

Complete the *insertionSort* function in the editor below.

insertionSort has the following parameter(s):

- *int arr[n]*: an array of integers

Returns

- *int*: the number of shifts required to sort the array

Input Format

The first line contains a single integer t , the number of queries to perform.

The following t pairs of lines are as follows:

- The first line contains an integer n , the length of *arr*.
- The second line contains n space-separated integers *arr*[i].

Constraints

- $1 \leq t \leq 15$
- $1 \leq n \leq 100000$
- $1 \leq a[i] \leq 10000000$

Sample Input

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

Sample Output

```
0
4
```

Explanation

The first query is already sorted, so there is no need to shift any elements. 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
```