6692 Lucky Number

Rock City traditionally holds an open-air non-reserved seat rock concert at the end of the summer. In the past, ticket holder would line up in a queue and enter the gate sequentially. Seating is on a first-in first-chose basis. So the i-th person in line will have $i-1$ less seats to chose from.

This year, the concert organiser wants to setup a priority system that still favours the ticket holder in the front of the queue, but also give those in the back a chance to enter the gate sooner. Each person will be assigned a priority number, which is his place in the queue plus a Lucky Number determined as follows.

On the ticket, there is a random positive number t. The number t is not unique, meaning that two or more people in queue can have the same number on their ticket. Suppose that the i-th person in the queue has the number t_i. The Lucky Number for person i is $j-i$ where the j-th person in the queue is the one farthest behind person i such that $t_j > t_i$.

In the following example, there are six people in line (from A to F). A has Lucky Number of 2 because C ($j=2$) is farthest behind A ($i=0$) such at $t_2 > t_0$. C has Lucky Number of 0 because no one behind him has a larger t_i.

Person	A	B	C	D	E	F
Position in queue	0	1	2	3	4	5
t_i	4	2	5	1	3	4
Lucky Number	2	4	0	2	1	0

Please write an efficient program to determine the largest Lucky Number in a given scenerial.

Technical Specification

1. t_i is the number printed on the ticket: $1 \le t_i \le 100,000$.
2. n is the number of people in the queue: $1 \le n \le 1,000,000$.

Input

The first line of the input contains an integer denoting the number of test cases to follow. For each test case, the first integer n denotes the number of people in the queue, followed by n integers, denoting $t_0, t_1, \ldots, t_{n-1}$.

Output

For each test case, output the largest Lucky Number.

Sample Input

```
2
6 4 2 5 1 3 4
7 9 8 5 4 3 3 3
```

Sample Output

```
4
0
```