The Athlete's True Score

Problem Description

Athletes are judged in many strange ways. Some lift heavy things, some run in circles, and some even throw pointy sticks. But in one particularly odd international competition, an athlete's greatness is determined by how often they manage to *improve* their performance over time.

The officials keep a record of the athlete's scores at successive events. From these scores, they define the athlete's **LIS-score**: the length of the *Longest Increasing Subsequence* (LIS) in the list of scores. That is, the largest number k such that there exist indices $i_1 < i_2 < \cdots < i_k$ with

$$\text{score}[i_1] < \text{score}[i_2] < \cdots < \text{score}[i_k].$$

Of course, the judges are picky. They don't just want to know the length of this subsequence, they also want you to print the actual indices of the chosen scores. And since judges often argue, you must break ties consistently: first choose the LIS whose last index is smallest. If there is still a tie, break it by the second-to-last index, and so on.

Your job is to help the officials by computing, for each test case, the LIS-score along with one such optimal subsequence of indices.

Input

The input consists of several test cases. Each test case begins with a single integer n ($1 \le n \le 10000$) on its own line, the number of recorded scores. The following line contains n integers, each between 0 and the maximum value of a standard C++ `int`, representing the scores.

The input terminates with a line containing -1; this line should not be processed.

Output

For each test case, output a single line in the following format:

$$k : i_1\, i_2\, \ldots\, i_k$$

where k is the LIS-score and $i_1, i_2, \ldots, i_k$ are the indices (0-based) of the chosen longest increasing subsequence. Ties are resolved as described above.

Samples

Sample Input 1

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

Sample Output 1

```
3: 0 2 4
```