Supernumbers in a permutation

An *n*-element permutation is an *n*-element sequence of distinct numbers from the set *{1, 2, ... n}*. For example the sequence 2, 1, 4, 5, 3 is a 5-element permutation. We are interested in the longest increasing subsequences in a permutation. In this exemplary permutation they are of length 3 and there are exactly 2 such subsequences: 2, 4, 5 and 1, 4, 5. We will call a number belonging to any of the longest increasing subsequences a *supernumber*. In the permutation 2, 1, 4, 5, 3 the supernumbers are 1, 2, 4, 5 and 3 is not a supernumber. Your task is to find all supernumbers for a given permutation.

Task

Write a program which

- reads a permutation from standard input,
- finds all its supernumbers,
- writes all found numbers to standard output.

Input

Ten test cases (given one under another, you have to process all!). Each test case consists of two lines. In the first line there is a number *n* ($1 \leq n \leq 100000$). In the second line: an *n*-element permutation - *n* numbers separated by single spaces.

Output

For every test case your program should write two lines. In the first line - the number of supernumbers in the input permutation. In the second line the supernumbers separated by single spaces in increasing order.

Example

Input:
5
2 1 4 5 3
[and 9 test cases more]

Output:
4
1 2 4 5
[and 9 test cases more]

Warning: large Input/Output data, be careful with certain languages