Problema InsertSort

Input file: `insertsort.in`
Output file: `insertsort.out`

Una dintre modalitățile de ordonare crescătoare a unui șir de numere, numită sortare prin inserare, este descrisă în continuare. Presupunem că elementele șirului sunt numere naturale, dispuse pe n poziții consecutive începând cu 1 și numite s_1, s_2, ..., s_n. Numim "prefix i" secvența de valori ce ocupă pozițiile 1, 2, ... i. Deci prefix 1 este primul element, prefix 2 este primul element urmat de al doilea, prefix n este tot șirul. La pasul i, cu i începând de la 2, presupunem că prefixul $i-1$ este ordonat și inserăm elementul s_i. Inserarea se face comparând elementul nou cu valori de la finalul prefixului $i-1$, în ordine descrescătoare a indicilor și atunci când elementul curent (cel aflat pe poziția i la începutul acestui pas) este mai mic, cele două se interschimbă. Evident, când se ajunge în față sau când se întâlnește un element mai mic decât cel curent, pasul curent se încheie, obținând astfel prefixul i ca fiind ordonat, fiind așadar pregătiți pentru pasul următor.

Iată un exemplu. Considerăm șirul $s_1 = 2$, $s_2 = 4$, $s_3 = 3$, $s_4 = 5$, $s_5 = 1$. La pasul 2 considerăm prefixul 1, format doar cu elementul cu valoarea 2, sortat. Inserăm pe $s_2 = 4$ în acest prefix. Nu este necesară nicio operație de interschimbare, așadar șirul rămâne neschimbat:

2 4 3 5 1.

La pasul 3, elementul $s_3 = 3$ trebuie inserat în prefixul sortat format din 2 și 4. El se compară cu 4, fiind nevoie de interschimbare, apoi întâlnește un element mai mic, 2 și ne oprim. Astfel, șirul devine:

2 3 4 5 1

La pasul 4 ajungem să inserăm pe $s_4 = 5$. Prefixul anterior este ordonat. Întrucât 5 este deja mai mare decât ultimul element al prefixului anterior sortat, nu este necesară nicio interschimbare.

La pasul 5, elementul 1 are nevoie de 4 interschimbări, în ordine cu 5, 4, 3, 2 până ajunge la locul său.

Cerința problemei este să determinăm câte elemente nu participă la nicio interschimbare dacă aplicăm acest algoritm de ordonare și care sunt aceste elemente (pentru exemplul de mai sus rezultatul ar fi fost 0 întrucât orice element trebuie supus cel puțin unei interschimbări).

Intrare

Pe prima linie a fișierului de intrare se găsește numărul n reprezentând dimensiunea șirului dat. Pe linia a 2-a se află cele n numere ale șirului. Se garantează că acestea sunt chiar valorile de la 1 la n, fiecare apărând exact o dată (altfel spus, o permutare a șirului 1, 2, ... n). Numerele de pe linia a 2-a sunt separate prin câte un spațiu.

Ieșire

Pe prima linie a fișierului de ieșire se va scrie r, numărul de valori care nu participă la nicio interschimbare. Pe linia a doua se află cele r valori determinate, în ordine crescătoare, separate prin câte un spațiu.

Restricții

- $1 \leq n \leq 100000$
- În materialele de specialitate se întâlnesc diverse variante de scriere a algoritmului de sortare prin inserare, inclusiv prin ocolirea operației de interschimbare a două valori. În această problemă analizăm strict varianta descrisă mai sus.

Exemplu

insertsort.in	insertsort.out
7 2 1 3 6 5 4 7	2 3 7