Problem A. A Pivotal Question

Source file name:	Apivotal.c, Apivotal.cpp, Apivotal.java, Apivotal.py
Input:	`Standard`
Output:	`Standard`

Quicksort is a recursive sorting algorithm developed in 1959 by Tony Hoare. One of the major steps in the algorithm is the *partition* step: given an element p in the array (the *pivot* element) rearrange the elements in the array as shown below where all the values in X_L are $\leq p$ and all elements in X_R are $> p$.

Figure 1 below shows an array before and after it's been partitioned with the pivot element 13. Note that the elements in X_L and X_R are typically not in sorted order and either one of them could be empty.

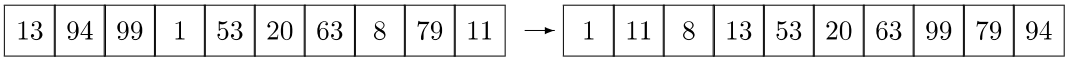

Figure 1: An array before and after a partition

How a partition is executed and how a pivot element is selected are fascinating questions but are not of interest to us. What we would like you to do is the following: given an array, determine all the values that could be the pivot value assuming the array has been partitioned, or determine that the array has not been partitioned.

Input

Input starts with a positive integer n ($1 \leq n \leq 10^6$) denoting the size of the array. Following this are n positive integers indicating the values in the array. All values are unique and $\leq 10^6$.

Output

Output m = the number of values in the array that could have served as pivot values to partition the array, followed by the pivot values in the order that they appear in the input. If $m > 100$ just output the first 100 of these pivot values. Note that a value of $m = 0$ indicates that the array is not partitioned.

Example

Input	Output
`10 1 11 8 13 53 20 63 99 79 94`	`3 1 13 63`