2966 - King's Quest

Europe - Northeastern - 2003/2004

Once upon a time there lived a king and he had *N* sons. And there were *N* beautiful girls in the kingdom and the king knew about each of his sons which of those girls he did like. The sons of the king were young and light-headed, so it was possible for one son to like several girls.

So the king asked his wizard to find for each of his sons the girl he liked, so that he could marry her. And the king's wizard did it - for each son the girl that he could marry was chosen, so that he liked this girl and, of course, each beautiful girl had to marry only one of the king's sons.

However, the king looked at the list and said: ``I like the list you have made, but I am not completely satisfied. For each son I would like to know all the girls that he can marry. Of course, after he marries any of those girls, for each other son you must still be able to choose the girl he likes to marry."

The problem the king wanted the wizard to solve had become too hard for him. You must save wizard's head by solving this problem.

Input

Input consists of several datasets. The first line of each dataset contains *N* - the number of king's sons ($1 \le N \le 2000$). Next *N* lines for each of king's sons contain the list of the girls he likes: first K_i - the number of those girls, and then K_i different integer numbers, ranging from 1 to *N* denoting the girls. The sum of all K_i does not exceed 200 000.

The last line of the input file contains the original list the wizard had made - *N* different integer numbers: for each son the number of the girl he would marry in compliance with this list. It is guaranteed that the list is correct, that is, each son likes the girl he must marry according to this list.

Output

Output *N* lines. For each king's son first print L_i - the number of different girls he likes and can marry so that after his marriage it is possible to marry each of the other king's sons. After that print L_i different integer numbers denoting those girls, in arbitrary order.

Sample Input

```
4
2 1 2
2 1 2
2 2 3
2 3 4
1 2 3 4
```

Sample Output

```
2 1 2
2 1 2
1 3
1 4
```