D. Merging Two Decks

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: input.txt
output: output.txt

There are two decks of cards lying on the table in front of you, some cards in these decks lay face up, some of them lay face down. You want to merge them into one deck in which each card is face down. You're going to do it in two stages.

The first stage is to merge the two decks in such a way that the relative order of the cards from the same deck doesn't change. That is, for any two different cards i and j in one deck, if card i lies above card j, then after the merge card i must also be above card j.

The second stage is performed on the deck that resulted from the first stage. At this stage, the executed operation is the turning operation. In one turn you can take a few of the top cards, turn all of them, and put them back. Thus, each of the taken cards gets turned and the order of these cards is reversed. That is, the card that was on the bottom before the turn, will be on top after it.

Your task is to make sure that all the cards are lying face down. Find such an order of merging cards in the first stage and the sequence of turning operations in the second stage, that make all the cards lie face down, and the number of turns is minimum.

Input

The first input line contains a single integer n — the number of cards in the first deck $(1 \le n \le 10^5)$.

The second input line contains n integers, separated by single spaces $a_1, a_2, ..., a_n$ $(0 \le a_i \le 1)$. Value a_i equals 0, if the i-th card is lying face down, and 1, if the card is lying face up. The cards are given in the order from the topmost one to the bottommost one.

The third input line contains integer m — the number of cards in the second deck $(1 \le m \le 10^5)$.

The fourth input line contains m integers, separated by single spaces $b_1, b_2, ..., b_m$ $(0 \le b_i \le 1)$. Value b_i equals 0, if the i-th card is lying face down, and 1, if the card is lying face up. The cards are given in the order from the topmost to the bottommost.

Output

In the first line print $n + m$ space-separated integers — the numbers of the cards in the order, in which they will lie after the first stage. List the cards from top to bottom. The cards from the first deck should match their indexes from 1 to n in the order from top to bottom. The cards from the second deck should match their indexes, increased by n, that is, numbers from $n + 1$ to $n + m$ in the order from top to bottom.

In the second line print a single integer x — the minimum number of turn operations you need to make all cards in the deck lie face down. In the third line print x integers: $c_1, c_2, ..., c_x$ $(1 \le c_i \le n + m)$, each of them represents the number of cards to take from the top of the deck to perform a turn operation. Print the operations in the order, in which they should be performed.

If there are multiple optimal solutions, print any of them. It is guaranteed that the minimum number of operations doesn't exceed $6 \cdot 10^5$.

Examples

input
3 1 0 1 4 1 1 1 1
output

```
1 4 5 6 7 2 3
3
5 6 7
```

input

```
5
1 1 1 1 1
5
0 1 0 1 0
```

output

```
6 1 2 3 4 5 7 8 9 10
4
1 7 8 9
```