Baltic OI '06 P5 - RLE Compression

Time limit: 5.0s **Memory limit:** 64M

RLE is a simple compression algorithm used to compress sequences containing subsequent repetitions of the same character. By compressing a particular sequence, we obtain its *code*. The idea is to replace repetitions of a given character (like `aaaaa`) with a counter saying how many repetitions there are. Namely, we represent it by a triple containing a repetition mark, the repeating character and an integer representing the number of repetitions. For example, `aaaaa` can be encoded as `#a5` (where `#` represents the repetition mark).

We need to somehow represent the alphabet, the repetition mark, and the counter. Let the alphabet consist of n characters represented by integers from the set $\Sigma = \{0, 1, \ldots, n-1\}$. The code of a sequence of characters from Σ is also a sequence of characters from Σ. At any moment, the repetition mark is represented by a character from Σ, denoted by e. Initially e is 0, but it may change during the coding. The code is interpreted as follows:

- any character a in the code, except the repetition mark, represents itself,
- if the repetition mark e occurs in the code, then the two following characters have special meaning:
  - if e is followed by ek, then it represents $k+1$ repetitions of e,
  - otherwise, if e is followed by $b0$ (where $b \neq e$), then b will be the repetition mark from that point on,
  - otherwise, if e is followed by bk (where $b \neq e$ and $k > 0$), then it represents $k+3$ repetitions of b.

Using the above scheme, we can encode any sequence of characters from Σ. For instance, for $n = 4$, the sequence `10022222233333030200 00` can be encoded as `10010230320100302101`. First character of the code `1` means simply `1`. Next `001` encodes `00`. Then, `023` represents `222222`, `032` represents `33333`, and `010` switches the repetition mark to `1`. Then `0302` represents itself and finally `101` encodes `0000`.

A sequence may be encoded in many ways and code length may vary. Given an already encoded sequence, your task is to find a code with the least number of characters.

Write a program that:

- Reads the size of the alphabet and the code of a sequence.
- Finds the shortest code for that sequence.
- Writes the result.

Input Specification

The first line contains one integer n $(2 \leq n \leq 100\,000)$: the size of the alphabet. The second line contains one integer m $(1 \leq m \leq 2\,000\,000)$: the length of the code. The last line contains m integers from the set $\{0, 1, \ldots, n-1\}$ separated by single spaces, representing the code of a sequence.

Output Specification

The first line should contain one integer m': the least number of characters in a code representing the given sequence. The last line of the output should contain m' integers from the set $\{0, 1, \ldots, n-1\}$ separated by single spaces: the code of the sequence. If there exist several shortest sequences, your program should output any one of them.

Sample Input 1

```
4
20
1 0 0 1 0 2 3 0 3 2 0 1 0 0 3 0 2 1 0 1
```

Sample Output 1

```
19
1 0 1 0 0 0 1 2 3 1 3 2 0 3 0 2 1 0 1
```

Sample Input 2

```
14
15
10 10 10 0 10 0 10 10 13 10 10 13 10 10 13
```

Sample Output 2

```
9
0 10 13 0 10 13 0 10 10
```