British Informatics Olympiad Final

6–8 April, 2001

Sponsored by Data Connection & e-competitions

Code Book

Alpha complex is large and sprawling, its many rooms and corridors designed to confuse the uninitiated. Its filing system is even worse. Several years ago they moved over to a new digital "it is not analogue, so it must be good" system. Every word that is used in filing reports has been encoded by a unique binary number, so reports just consist of long sequences of `0`s and `1`s. The system has three saving graces:

- The encoding is unambiguous; i.e. no encoded word is the prefix of another encoded word. For example, if `010` is a word there are can be no other words beginning with `010` and the words `0` and `01` will not exist.
- The lexigraphic order of the words has been maintained. In other words, if $word_1$ is before $word_2$ in the dictionary, the encoding of $word_1$ would also be before the encoding of $word_2$. (In the lexigraphic ordering of numbers they are aligned on the left, so `10001` is before `101` but after `100`.)
- The encoding is optimal. If you counted how many `0`s and `1`s are used in total over all the words in all the files, it would not be possible to lower this number by using a different encoding that obeys the previous two rules.

For example, if there were three words that were encoded, the first (lexigraphically) occurring 5 times, the second 8 and the third 6 the encoding would be `00`, `01` and `1`. This has and overall total of $5 \times 2 + 8 \times 2 + 6 = 32$. Another encoding that meets the first two conditions would be `0`, `10` and `11`, but this has a score of 33 so does not meet the final condition. The encoding `00`, `1` and `01`, which would have a score of 30, does not meet the second condition. The encoding `10`, `0` and `1` (with a score of 24) does not meet the first condition since `10` is ambiguous (is it `10` or `1` then `0`?)

Write a program that, given a list of word frequencies (corresponding to words in lexigraphic order) calculates an encoding that meets all three conditions. The first line of input will consist of a single integer, n ($1 \leq n \leq 100$), indicating the number of frequencies to follow. The next n lines will each consist of a single number (between 1 and 100 inclusive), the i^{th} line indicating the frequency of the i^{th} word in lexigraphic order.

You should output n lines. The i^{th} line should consist of your encoding for the i^{th} word.

Sample Input

```
3
5
8
6
```

Sample Output

```
00
01
1
```