Passing Notes

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

This is a communication problem.

Bob and Alice played the guessing game n times. After correctly guessing the number Alice was thinking of each time, he would write down a 01-string of length 8 on a note to record that number, resulting in a total of n notes.

However, if the note is flipped, the recorded 01-string will also be reversed, making it unclear whether the string should be read from left to right or from right to left. To avoid ambiguity, Bob must design an encoding and decoding scheme so that regardless of whether the note is flipped, the original number can be uniquely restored from the information on the note.

After some time, Bob finds the n notes again, and he needs to restore the recorded numbers based on the information on the notes.

Your program will run **twice** for each test case:

- **First Run**: Bob and Alice played the guessing game n times. After correctly guessing the number Alice was thinking of each time, he records that number by writing a 01-string of length 8 on a note.
- **Second Run**: Bob finds the n notes again and needs to restore the recorded numbers based on the information on the notes. These notes are exactly what was recorded during the first run. Each note may have been flipped (i.e., its 01-string may have been reversed), and the order of the notes may have been scrambled. Furthermore, the second run will not receive any additional information from the first run. If your program correctly restores all the numbers recorded on the notes, it passes this test case.

Input

The first line of input contains two integers op ($op \in \{1, 2\}$) and n ($1 \leq n \leq 100$), representing the run number of the program and the number of notes, respectively. It is guaranteed that n remains consistent between the two runs.

If $op = 1$, your program needs to write notes to record the numbers. The next n lines each contain an integer x ($0 \leq x \leq 100$), representing the number to be recorded.

If $op = 2$, your program needs to restore the numbers recorded on the notes. The next n lines each contain a 01-string of length 8, representing the information on the notes. These n 01-strings will be given in **any order** and do not necessarily correspond to the order of the numbers given in the first run.

Output

If $op = 1$, output n lines, each containing a 01-string of length 8, representing the information recorded on the notes.

If $op = 2$, output n lines. For each of the n 01-strings given in the input, output a line containing an integer that represents the number restored from the information on the notes.

Examples

standard input	standard output
1 2 0 100	00000000 00001111
2 2 00001111 00000000	100 0

Note

The example demonstrates a certain solution running twice on the sample test data.