Problem A
Arranging Hat

Arranging Hat is a cushy job indeed; high impact work, absolute authority, and 364 days of holiday every year. However, the hat has decided that it can do even better—it would like very much to become a tenured professor.



The Arranging Hat. Image by Lisa Abose

Recently the hat has been reading computer science papers in its ample spare time, and of course, being an arranging hat, it is particularly interested in learning more about sorting algorithms.

The hat's new contribution is to a class of algorithms known as *lossy* sorting algorithms. These usually work by removing some of the input elements in order to make it easier to sort the input (e.g., the Dropsort algorithm), instead of sorting all the input.

The hat is going to go one better—it is going to invent a lossy sorting algorithm for numbers that does not remove any input numbers and even keeps them in their original place, but instead changes some of the digits in the numbers to make the list sorted.

The lossiness of the sorting operation depends on how many digits are changed. What is the smallest number of digits that need to be changed in one such list of numbers, to ensure that it is sorted?

Input

The input consists of:

- one line containing the integers n and m ($1 \leq n \leq 40$, $1 \leq m \leq 400$), the number of numbers and the number of digits in each number, respectively.
- n lines each containing an integer v ($0 \leq v < 10^m$). The numbers are zero-padded to exactly m digits.

Output

Write a sorted version of the array, after making a minimum number of digit changes to make the numbers sorted (the numbers must remain zero-padded to m digits). If there are multiple optimal solutions, you may give any of them.

Sample Input 1	Sample Output 1
5 3 111 001 000 111 000	001 001 001 111 200

Sample Input 2	Sample Output 2
15 3 999 888 777 666 555 444 333 222 111 222 333 444 555 666 999	199 288 377 466 555 644 733 822 911 922 933 944 955 966 999