

Task 1

One measure of "unsortedness" in a sequence is the number of pairs of entries that are out of order with respect to each other. For instance, in the letter sequence "DAABEC", this measure is 5, since D is greater than four letters to its right and E is greater than one letter to its right. This measure is called the number of inversions in the sequence. The sequence "AACEDGG" has only one inversion (E and D)--it is nearly sorted--while the sequence "ZWQM" has 6 inversions (it is as unsorted as can be--exactly the reverse of sorted).

You are responsible for cataloguing a sequence of DNA strings (sequences containing only the four letters A, C, G, and T). However, you want to catalog them, not in alphabetical order, but rather in order of "sortedness", from "most sorted" to "least sorted". All the strings are of the same length.

Input

The first line of the input is an integer M , then a blank line followed by M datasets. There is a blank line between datasets. The first line of each dataset contains two integers: a positive integer n ($0 < n \leq 50$) giving the length of the strings; and a positive integer m ($0 < m \leq 100$) giving the number of strings. These are followed by m lines, each containing a string of length n .

Output

For each dataset, output the list of input strings, arranged from "most sorted" to "least sorted". If two or more strings are equally sorted, list them in the same order they are in the input file. Print a blank line between consecutive test cases.

Sample Input

Sample Input	Output for Sample Input
6 10 6 AACATGAAGG TTTTGGCCAA TTTGGCCAAA GATCAGATTT CCCGGGGGGA ATCGATGCAT 7 5 CTGAGCC GCCACGG TATGCGT ATCTCGT GGCTGTT 7 2 GTAATCG AGGATAG 5 5 AGATC TATGT AACTC TCCTC AGCGC 1 3 C T T 6 3 CCAAGC TTATGA AACCCC	CCCGGGGGGA AACATGAAGG GATCAGATTT ATCGATGCAT TTTTGGCCAA TTTGGCCAAA GGCTGTT ATCTCGT GCCACGG TATGCGT CTGAGCC AGGATAG GTAATCG AACTC AGATC TATGT AGCGC TCCTC C T T AACCCC CCAAGC TTATGA

Task 2

Andy, 8, has a dream - he wants to produce his very own dictionary. This is not an easy task for him, as the number of words that he knows is, well, not quite enough. Instead of thinking up all the words himself, he has a brilliant idea. From his bookshelf he would pick one of his favorite story books, from which he would copy out all the distinct words. By arranging the words in alphabetical order, he is done! Of course, it is a really time-consuming job, and this is where a computer program is helpful.

You are asked to write a program that lists all the different words in the input text. In this problem, a word is defined as a consecutive sequence of alphabets, in upper and/or lower case. Words with only one letter are also to be considered. Furthermore, your program must be Case Insensitive. For example, words like "Apple", "apple" or "APPLE" must be considered the same.

Input

The input file is a text with no more than 5000 lines. An input line has at most 200 characters. Input is terminated by EOF.

Output

Your output should give a list of different words that appears in the input text, one in a line. The words should all be in lower case, sorted in alphabetical order. You can be sure that the number of distinct words in the text does not exceed 5000.

Sample Input

Adventures in Disneyland

Two blondes were going to Disneyland when they came to a fork in the road. The sign read: "Disneyland Left."

So they went home.

Sample Output

**a
adventures
blondes
came
disneyland
fork
going
home
in
left
read
road
sign
so
the
they
to
two
went
were
when**

Task 3

Implement Quick Sort algorithm for numbers. (After solving task 1 & 2)