

Round 2 2014

[A. Data Packing](#)[B. Up and Down](#)[C. Don't Break The Nile](#)**[D. Trie Sharding](#)**[Contest Analysis](#)[Questions asked](#)

Submissions

Data Packing

5pt	Not attempted 2522/2552 users correct (99%)
8pt	Not attempted 2454/2525 users correct (97%)

Up and Down

7pt	Not attempted 1728/2380 users correct (73%)
11pt	Not attempted 1351/1426 users correct (95%)

Don't Break The Nile

10pt	Not attempted 685/1167 users correct (59%)
20pt	Not attempted 221/287 users correct (77%)

Trie Sharding

9pt	Not attempted 1408/1533 users correct (92%)
30pt	Not attempted 78/115 users correct (68%)

Top Scores

Gennady.Korotkevich	100
yeputons	100
squark	100
wata	100
rng..58	100
PavelKunyavskiy	100
ecnerwala	100
winger	100
WJMZBMR	100
eatmore	100

Problem D. Trie Sharding

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the [Quick-Start Guide](#) to get started.

Small input
9 points

Solve D-small

Large input
30 points

Solve D-large

Problem

A set of strings **S** can be stored efficiently in a *trie*. A trie is a rooted tree that has one node for every prefix of every string in **S**, without duplicates.

For example, if **S** were "AAA", "AAB", "AB", "B", the corresponding trie would contain 7 nodes corresponding to the prefixes "", "A", "AA", "AAA", "AAB", "AB", and "B".

I have a server that contains **S** in one big trie. Unfortunately, **S** has become very large, and I am having trouble fitting everything in memory on one server. To solve this problem, I want to switch to storing **S** across **N** separate servers. Specifically, **S** will be divided up into disjoint, non-empty subsets **T₁**, **T₂**, ..., **T_N**, and on each server *i*, I will build a trie containing just the strings in **T_i**. The downside of this approach is the total number of nodes across all **N** tries may go up. To make things worse, I can't control how the set of strings is divided up!

For example, suppose "AAA", "AAB", "AB", "B" are split into two servers, one containing "AAA" and "B", and the other containing "AAB", "AB". Then the trie on the first server would need 5 nodes ("", "A", "AA", "AAA", "B"), and the trie on the second server would also need 5 nodes ("", "A", "AA", "AAB", "AB"). In this case, I will need 10 nodes altogether across the two servers, as opposed to the 7 nodes I would need if I could put everything on just one server.

Given an assignment of strings to **N** servers, I want to compute the worst-case total number of nodes across all servers, and how likely it is to happen. I can then decide if my plan is good or too risky.

Given **S** and **N**, what is the largest number of nodes that I might end up with? Additionally, how many ways are there of choosing **T₁**, **T₂**, ..., **T_N** for which the number of nodes is maximum? Note that the **N** servers are different -- if a string appears in **T_i** in one arrangement and in **T_j** (*i* != *j*) in another arrangement, then the two arrangements are considered different. Print the remainder of the number of possible arrangements after division by 1,000,000,007.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case starts with a line containing two space-separated integers: **M** and **N**. **M** lines follow, each containing one string in **S**.

Output

For each test case, output one line containing "Case **#i**: **X Y**", where **i** is the case number (starting from 1), **X** is the worst-case number of nodes in all the tries combined, and **Y** is the number of ways (modulo 1,000,000,007) to assign strings to servers such that the number of nodes in all **N** servers are **X**.

Limits

$1 \leq T \leq 100$.

Strings in **S** will contain only upper case English letters.

The strings in **S** will all be distinct.

$N \leq M$

Small dataset

$1 \leq M \leq 8$

$1 \leq N \leq 4$

Each string in **S** will have between 1 and 10 characters, inclusive.

Large dataset

$1 \leq M \leq 1000$

$1 \leq N \leq 100$

Each string in **S** will have between 1 and 100 characters, inclusive.

Sample

Input	Output
2	Case #1: 10 8
4 2	Case #2: 7 30
AAA	
AAB	
AB	
B	
5 2	
A	
B	
C	
D	
E	

All problem statements, input data and contest analyses are licensed under the [Creative Commons Attribution License](#).

© 2008-2017 Google [Google Home](#) - [Terms and Conditions](#) - [Privacy Policies and Principles](#)

Powered by



Google Cloud Platform