# Search History

Team Id : 39
Team member : B05902043 B05902045 B0902129

## 1, Topic

Trie

## 2, Description

The search history is most people's secret. We've all viewed a website or two that we'd prefer to keep hidden from prying eyes. Some might wisely use incognito window/private browsing, while others just leaving those in their search record. One day, you find your best friend Lawrence's personal computer left on his desk, unlocked. You want to find out if Lawrence has some nasty little secret so you can pull a prank on him. You search in Lawrence's search history to see if there is anything interesting. You are surprised to find out that if you enter an incomplete word, the search engine will offer you several word suggestions which start with the prefix you entered. Try to simulate the algorithms.

Note:
We are searching for records so there will be same record appearing more than once. (If you are so fond of that keyword) But we just want one suggestion(result) for each repeating record.
***The input dataset might have repeating words but you only have to print it out once.***

It's not as hard as HW2_Information Retrieval so don't be afraid. You can practice the useful data structure ***trie***.

suggested BGM while coding:
Nightcore-you don't have to try https://www.youtube.com/watch?v=x9WpVoAC3tk

**3, Input and output format**

**Input:**

$N$

$s_1$

$s_2$

$s_3$

…

$s_n$

$M$

$q_1$

$q_2$

…

$q_m$

**Output:**

$a1_1$

$a1_2$

…

$a1_i$

$a2_1$

$a2_2$

…

$a2_i$

…

**Input format:**

N: Number of the words in the dataset

$s_i$ : the data string

M: Number of the queries

$q_i$ : the query string

All the strings are composed of lower case

20% : 1 <= N, M <= 100

40% : 1 <= N, M <= 10000

40% : 1 <= N, M <= 100000

**Output format:**
You have to output all the strings which start with the given prefix(query). If there are more than one answer, print them on the ***Lexicographical order***.
For example, ***goo*** is the prefix of ***good*** , ***goodnight*** and ***goose***, so you have to output ***good*** , ***goodnight*** and ***goose***. If there doesn't exist any strings that start with the given prefix(query), output "NO MATCHING!".

**4, Sample input and output**
**Input:**
12
goose
good
happy
goose
goose
apple
app
hot
god
goodnight
goodnight
hahaha
2
goo
hhh
**Output:**
good
goodnight
goose
NO MATCHING!
**5, Time and memory limits**
Time : 1sec
Memory : 256MB