Задача A. Лингвист - неудачник

Имя входного файла:	`suffix.in`
Имя выходного файла:	`suffix.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

В древние времена юный лингвист составил новое слово. Поскольку у него было ещё много времени, он решил проделать со своим словом странную вещь: поскольку его слово это строка, он выписал все суффиксы этой строки и пронумеровал их по порядку от 1 до n (самый короткий суффикс имеет номер n, самый длинный — номер 1). Затем он расположил все суффиксы в алфавитном порядке, и записал получившуюся последовательность чисел (каждое число соответствует номеру суффикса). Назовём эту операцию построением **суффиксного массива**.

Но на следующее утро случилась страшная беда: младший брат лингвиста порвал бумажку с записанным словом на отдельные листочки, чтобы на каждом было по одной букве, а затем расположил все листочки в случайном порядке, так что получилось новое слово. Лингвист ужаснулся, ведь он не помнил, как именно выглядело его слово, а он уже сообщил другим лингвистам о нём и они вот-вот придут. Лингвист уверен, что если он создаст из листочков новое слово, у которого будет тот же **суффиксный массив**, то он обязательно получит своё старое слово, однако времени почти не осталось. Помогите бедному лингвисту не опозориться перед своими коллегами и восстановить старое слово.

Формат входных данных

В первой строке входных данных содержится длина слова n ($1 \leqslant n \leqslant 500$).

В второй строке содержится перестановка символов длины n, из которых состояло исходное слово лингвиста.

В третьей строке содержится перестановка из n чисел, записанных через пробел - **суффиксный массив** исходной строки.

Формат выходных данных

Выведите исходное слово или, если лингвист ошибся и невозможно составить из данных букв слово с данным **суффиксным массивом**, то выведите '`No`' без кавычек.

Примеры

suffix.in	suffix.out
7 babaaca 7 5 1 3 6 2 4	abacaba
6 kekkek 6 2 3 1 4 5	No

Задача B. Множественный поиск

Имя входного файла:	`search4.in`
Имя выходного файла:	`search4.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

Дано множество строк S и строка t. Требуется для каждой сроки $p \in S$ определить, встречается ли она в t как подстрока.

Формат входных данных

Первая строка входного файла содержит целое число n — мощность S ($1 \leqslant n \leqslant 10^6$). Следующие n строк содержат по одной строке из S. Сумма длин всех строк из S не превосходит 10^6. Последняя строка входного файла содержит t ($1 \leqslant t \leqslant 10^6$). Все строки состоят из прописных латинских букв.

Формат выходных данных

Для каждой сроки из S выведите «`YES`», если она встречается в t и «`NO`» в противном случае. Строки нумеруются в порядке появления во входном файле.

Примеры

search4.in	search4.out
3 abc abcdr abcde xabcdef	YES NO YES

Задача C. Вирусы

Имя входного файла:	`virus.in`
Имя выходного файла:	`virus.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Комитет По Исследованию Бинарных Вирусов обнаружил, что некоторые последовательности единиц и нулей являются кодами вирусов. Комитет изолировал набор кодов вирусов. Последовательность из единиц и нулей называется безопасной, если никакой ее подотрезок (т.е. последовательность из соседних элементов) не является кодом вируса. Сейчас цель комитета состоит в том, чтобы установить, существует ли бесконечная безопасная последовательность из единиц и нулей.

Формат входных данных

Первая строка входного файла `virus.in` содержит одно целое число N, равное количеству всех вирусных кодов. Каждая из следующих n строк содержит непустое слово, составленное из символов `0` и `1` — код вируса. Суммарная длина всех слов не превосходит 30000.

Формат выходных данных

Первая и единственная строка выходного файла должна содержать слово:

- TAK — если бесконечная, безопасная последовательность из нулей и единиц сушествует;
- NIE — в противном случае.

Примеры

virus.in	virus.out
3 01 11 00000	NIE
3 011 11 0000	TAK

Задача D. Подозрительные строки

Имя входного файла:	`strings.in`
Имя выходного файла:	`strings.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вы работаете в компании, специализирующейся в технологиях, связанных с интернетом, и ваш текущий проект — спам-фильтр. Фильтр определяет, содержит ли строка спам, используя *словарь спам-слов*. Если в строке содержится хотя бы одно слово из этого словаря как подстрока, фильтр считает, что исходная строка подозрительна.

Вы стали решать более интересную задачу: сколько существует различных подозрительных строк длины n, состоящих из строчных букв латинского алфавита для данного словаря спам-слов. Найдите ответ по модулю 10 000.

Формат входных данных

В первой строке содержатся два числа n и k ($1 \leqslant n < 2^{31}, 1 \leqslant t \leqslant 10$) — требуемая длина слов и количество слов в словаре спам-слов соотвественно.

Следующие k строк являются строками словаря. Гарантируется, что они состоят из строчных латинских букв, они не пустые, и их длина не превосходит 10 символов.

Формат выходных данных

Выведите ответ по модулю 10 000.

Примеры

strings.in	strings.out
1 1 x	1
2 2 ab bb	2
5 2 ab bb	6350
5 2 aab bba	4054
5 9 xxxxxx xxx x yyxyy xxxyxxx y yx xy zzzzzzzzzz	8752
2147483647 10 aaaaaaaaaa bbbbbbbbbb cccccccccc dddddddddd eeeeeeeeee ffffffffff gggggggggg hhhhhhhhhh xxxxxxxxxx zzzzzzzzzz	5040