Задача A. Три буквы

Имя входного файла:	`abc.in`
Имя выходного файла:	`abc.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Напомним, что строка $B = b_1b_2b_3\dots b_m$, является подпоследовательностью строки $A = a_1a_2a_3\dots a_n$, если существует строго возрастающая последовательность $\{i_1, i_2, i_3, \dots, i_m\}$ индексов A, такая, что для всех $j \in \{1, \dots, m\}$, выполняется $A_{i_j} = B_j$. Например, $B =$ "aba" является подпоследовательностью строки $A =$ "$abacaba$". Последовательность индексов в этом случае может быть такой: $\{1, 2, 3\}$.

Пусть Вам дана строка S состоящая только из маленьких букв латинского алфавита. Ваша задача заключается в том, чтобы посчитать количество ее подпоследовательностей «`abc`».

Формат входного файла

В единственной строке входного файла записана строка S, длиной не более 100 000 символов.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Пример

abc.in	abc.out
abc	1
ab	0

Задача B. Все, что движется

Имя входного файла:	`moving.in`
Имя выходного файла:	`moving.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дано поле M на N и объекты на нем в моменты времени T и $T+1$.

Каждый объект представлен одной латинской буквой (регистр важен) и в любой момент времени может занимать ровно одну клетку поля.

Объект движется, если в два последовательных момента времени его положения различаются.

Найдите все, что движется.

Формат входного файла

В первой строке входного файла — два целых числа M, N, $1 \le M, N \le 100$.

В следующих M строках по N символов — поле в момент времени T. Каждая символ либо является точкой «.», и это означает, что в этом месте поля никого нет, либо латинская буква, обозначающая то, что в этом месте находится объект. Никакие два объекта не обозначены одним и тем же символом.

Далее находится пустая строка.

В следующих M строках находится описание того же поля в момент времени $T+1$ в том же формате. Множество объектов, находящихся на поле в момент времени T, равно множеству объектов в момент времени $T+1$.

Формат выходного файла

В первой строке выведите количество движущихся объектов.

Во второй строке выведите символы, соответствующие движущимся объектам, в алфавитном порядке, причем сначала выведите все маленькие латинские буквы, затем все большие. Пробелы между символами выводить не надо.

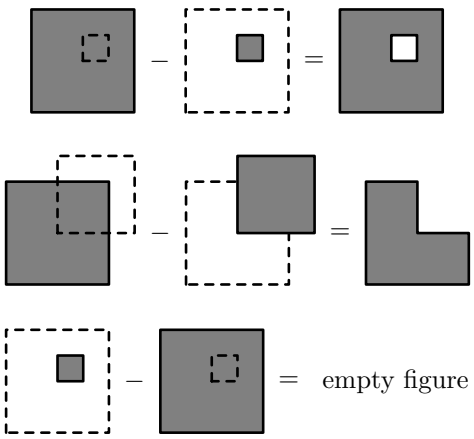
Примеры

moving.in	moving.out
2 2 .A .. A. ..	1 A
3 3 x.O .X. .o. x.O .X. .o.	0

Задача C. Развлечение с квадратиками

Имя входного файла:	`squares.in`
Имя выходного файла:	`squares.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

У Димы есть n квадратиков, стороны которых параллельны осям координат. Для двух квадратов A и B обозначим фигуру, которая состоит из точек A, не принадлежащих B, как $A - B$.



Теперь Дима хочет найти количество наборов из четырех различных квадратиков A, B, C и D, таких что $A-B$ и $C-D$ равны с точностью до параллельного переноса (повороты не разрешаются).

Формат входного файла

Первая строка входного файла содержит n — количество квадратиков ($4 \le n \le 400$). Следующие n строк описывают квадратики. Каждый квадратик описывается тремя целыми числами: x, y и l — координатами левого нижнего угла и длиной стороны ($-10^9 \le x, y \le 10^9$, $1 \le l \le 10^9$).

Формат выходного файла

Выведите количество наборов из четырех различных квадратиков A, B, C и D, таких что $A-B$ и $C-D$ равны с точностью до параллельного переноса. Наборы, содержащие одно и то же множество квадратиков, но в разном порядке, считаются различными.

Пример

squares.in	squares.out
4 0 0 3 1 1 1 2 2 1 1 1 3	6
4 0 0 3 1 1 3 2 2 3 4 4 3	0

В первом примере подходят следующие наборы: $\langle 0, 1, 3, 2\rangle$, $\langle 1, 0, 2, 3\rangle$, $\langle 1, 3, 2, 0\rangle$, $\langle 2, 0, 1, 3\rangle$, $\langle 2, 3, 1, 0\rangle$, $\langle 3, 2, 0, 1\rangle$.

Задача D. Морфизм

Имя входного файла:	`morpher.in`
Имя выходного файла:	`morpher.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Рассмотрим слова, состоящие из первых n букв латинского алфавита. *Морфизм* — это функция f, которая по букве возвращает слово. Рассмотрим пример морфизма: $f(\texttt{A}) = \texttt{ABC}$, $f(\texttt{B}) = \texttt{A}$, $f(\texttt{C}) = \texttt{BC}$.

Если мы рассмотрим слово $w = c_1c_2\ldots c_l$ и применим к нему морфизм f, мы получим слово $f(w) = f(c_1)f(c_2)\ldots f(c_l)$. Например, для морфизма из предыдущего параграфа $f(\texttt{ABC}) = \texttt{ABCABC}$.

Мы можем применять морфизм к слову несколько раз. Положим $f^0(w) = w$, и для $k > 0$ положим $f^k(w) = f(f^{k-1}(w))$.

По заданному морфизму f, слову w, числу k и числу p, найдите p-й символ слова $f^k(w)$.

Формат входного файла

Первая строка входного файла содержит числа n, k и p ($1 \le n \le 10$, $0 \le k \le 10^9$, $1 \le p \le 20$). Вторая строка входного файла содержит слово w. Его длина не превышает 50. Следующие n строк содержит $f(\texttt{A})$, $f(\texttt{B})$, и т.д. Каждое значение — это строка, содержащая от 1 до 50 символов.

Формат выходного файла

Выведите один символ — p-й символ $f^k(w)$, или "`-`" если такой символ отсутствует.

Пример

morpher.in	morpher.out
3 1 5 ABC ABC A BC	B
3 1 7 ABC ABC A BC	-