Задача A. Суффиксное дерево

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

Дана строка s. Постройте сжатое суффиксное дерево для строки s и выведите его. Найдите такое дерево, которое содержит минимальное количество вершин.

Формат входных данных

В первой строке записана строка s ($1 \leqslant |s| \leqslant 10^5$), последний символ строки доллар «$», остальные символы строки маленькие латинские буквы.

Формат выходных данных

Пронумеруйте вершины дерева от 0 до $n-1$ в порядке обхода в глубину, обходя поддеревья в порядке лексикографической сортировки исходящих из вершины рёбер. Используйтся ASCII-коды символов для определения их порядка.

В первой строке выведите число n – количество вершин дерева. В следующих $n-1$ строках выведите описание вершин дерева, кроме корня, в порядке увелечения их номеров.

Описание вершины дерева v состоит из трёх целых чисел: p, lf, rf, где p ($0 \leqslant p \leqslant n, p \neq v$) – номер родителя текущей вершины. На ребер ведущем из p в v написана подстрока $s[lf..rf)$ ($0 \leqslant lf < rf \leqslant |s|$).

Примеры

suftree.in	suftree.out
aaa$	7 0 3 4 0 0 1 2 3 4 2 1 2 4 3 4 4 2 4
b$	3 0 1 2 0 0 2
ababa$	10 0 5 6 0 0 1 2 5 6 2 1 3 4 5 6 4 3 6 0 1 3 7 5 6 7 3 6

Замечание

В этой задаче запрещается строить суфдерево через другие структуры данных.

Задача B. Динамический поиск подстрок

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	6 секунд
Ограничение по памяти:	256 мегабайт

Словарь— это множество слов. Вы должны уметь обрабатывать запросы трех типов:

- «`+ word`» — добавить слово `word` в словарь, если оно в нем не присутствует.
- «`- word`» — удалить слово `word` из словаря, если оно там присутствует.
- «`? text`» — вычислить суммарное количество вхождений всех слов из словаря в текст `text`, при этом, если слово входит в текст несколько раз, то необходимо учесть каждое вхождение.

Гарантируется, что любое слово или текст являются непустыми строками, состоящими из букв `a`, `b` и `c`, суммарная длина которых не превосходит L. Однако, для упрощения задачи перед выполнением каждого запроса необходимо поступить следующим образом: пусть x обозначает ответ на последний запрос `?`, или 0, если таких запросов еще не было. Тогда необходимо очередную строку (`word` или `text`) циклически сдвинуть x раз. Напомним, что циклическим сдвигом строки $s = s_0 s_1 \dots s_{|s|}$ называется строка $s' = s_1 \dots s_{|s|} s_0$.

Формат входных данных

В первой строке дано одно число Q — число запросов. В следующих Q строках находятся запросы. Суммарная длина строк во всех запросах не превосходит L ($L \leqslant 5\,000\,000$)

Формат выходных данных

Для каждого запроса «`?`» выведите одно число — ответ на него.

Примеры

стандартный ввод	стандартный вывод
11 + a + a - a - ab ? abca + ab + a ? abaaabb ? aaabbab + baa ? babaca	0 6 5 7

Задача C. Помогите, спасите!

Имя входного файла:	`stdin`
Имя выходного файла:	`stdout`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Дана строка. Найдите для каждого её префикса количество различных подстрок в нём.

Формат входных данных

В единственной строке входных данных содержится непустая строка S, состоящая из N ($1 \leqslant N \leqslant 2 \cdot 10^5$) маленьких букв английского алфавита.

Формат выходных данных

Выведите N строк, в i-й строке должно содержаться количество различных подстрок в i-м префиксе строки S.

Примеры

stdin	stdout
aabab	1 2 5 8 11
atari	1 3 5 9 14

Задача D. LZSS encoding

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

Алиса хочет отправить сообщение Бобу. Она хочет зашифровать сообщение, используя оригинальный метод шифрования. Сообщение – строка S, состоящая из N строчных английских букв.

$S[a \dots b]$ означает подстроку S от $S[a]$ до $S[b]$ ($0 \leqslant a \leqslant b < N$). Если первые i букв уже зашифрованы, Алиса найдёт такие (j, k): $s[j..j+k] = s[i..i+k], k \geqslant 0, 0 \leqslant j < i, k = \max$. Если несколько j дают максимальное k, Алиса выберет минимальное j. Если $k > 0$ Алиса добавит пару $\langle j, k \rangle$ в шифр и увеличит i на k, иначе Алиса добавит -1 и ASCII код буквы $S[i]$ в шифр и увеличит i на 1.

Очевидно шифр начнёт с -1, далее будет ASCII код символа $S[0]$. Помогите Алисе реализовать её метод шифрования.

Формат входных данных

Перрвая строка ввода содержит количество тестов T ($1 \leqslant T \leqslant 50$). Следующие T строк содержат сообщения для шифровки, каждое длины от 1 до 10^5, состоящие из строчных английских букв. Гарантируется, что суммарная длина всех сообщений не превосходит $2 \cdot 10^6$.

Формат выходных данных

Для каждого теста на отдельной строке выведите "`Case #X:`", где X – номер теста, нумерация с 1. Далее выведите шифр, в каждой строке по два целых числа через пробел.

Пример

lzss.in	lzss.out
2 aaaaaa aaaaabbbbbaaabbc	Case #1: -1 97 5 0 Case #2: -1 97 4 0 -1 98 4 5 5 2 -1 99

Замечание

Здесь и в следующих задачах запрещается использовать суффиксный массив для решения задач. Если вас переполняет ненависть к суффиксному дереву, можно использовать суфавтомат

Задача E. Общие префиксы

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	4 секунды
Ограничение по памяти:	1024 мегабайта

Обозначим $LCP(X, Y)$ за длину наибольшего общего префикса строк X и Y. Например $LCP(\text{"}aboba\text{"}\,\text{"}abaca\text{"}) = 2$, а $LCP(\text{"}aaa\text{"}\,\text{"}aaaa\text{"}) = 3$.

Дана строка S длины N. Обозначим также суффикс строки S, начинающийся с i-го символа за S_i. Для всех $k = 1, \ldots, n$ Найдите сумму $LCP(S_k, S_1) + LCP(S_k, S_2) + \ldots + LCP(S_k, S_n)$

Формат входных данных

В первой строке вводится число N ($1 \leqslant N \leqslant 10^6$)

Во второй записана строка S, состоящая из малых букв английского алфавита

Формат выходных данных

Выведите N чисел, по одной в каждой строке — значение суммы для $k = 1, \ldots, n$

Примеры

стандартный ввод	стандартный вывод
3 abb	3 3 2
11 mississippi	11 16 14 12 13 11 9 7 4 3 4

Задача F. Игры с Укконеном

Имя входного файла:	`ukkonen.in`
Имя выходного файла:	`ukkonen.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	512 мегабайт

На этот раз коварный Ишма Дровкин решил сорвать лекцию параллели А про алгоритм Укконена, позволяющий построить сжатое суффиксное дерево за линейное время, заявившись на неё с правилами новой игры. К счастью, эту лекцию вел сам завуч ЛКШ Станкей Андревич, который рассудил, что новая игра может помочь школьникам в изучении непростого алгоритма. Правила игры заключались в следующем.

В самом начале два участвующих в игре школьника загадывают некоторую строку S, состоящую из строчных букв латинского алфавита. После этого они по очереди делают ходы, каждый из которых как-то меняет вторую строку T, которая изначально пуста. При этом первый школьник своим ходом должен дописать ровно один символ в конец строки T, а второй — в ее начало. Главное правило игры заключается в том, что после каждого хода строка T должна оставаться подстрокой строки S. Соответственно, школьник, который не сможет сделать ход, соблюдающий это правило, объявляется проигравшим.

Пока школьники развлекались, играя в эту игру, Станкей Андревич заметил, что в ситуации, когда оба игрока играют оптимально, победителя всегда можно определить по строке S, загаданной школьниками в самом начале. Чтобы убить оставшееся от лекции время, он решил написать программу, умеющую решать эту задачу. А поскольку завуч ЛКШ никогда и ничего не должен делать сам, вам необходимо сделать это за него.

Формат входных данных

Первая строка входного файла содержит одно натуральное число N — количество строк, которые хочет изучить Станкей Андревич. Следующие N строк содержат изучаемые строки.

Все изучаемые строки состоят только из строчных букв английского алфавита, а их суммарная длина не превышает $2 \cdot 10^3$.

Формат выходных данных

В выходной файл выведите слово «`First`», если победу при этой строке S одержит школьник, делающий первый ход в игре, и «`Second`» — в противном случае.

Примеры

ukkonen.in	ukkonen.out
2 aaabaaa aaaba	First Second