Задача A. Строки

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Вася посещает занятия по программированию. К сожалению, недавно он заболел и не смог прийти на лекцию. Он смог узнать лишь, что на лекции проходили поиск подстроки в строке. Вася боится, что на следующей тренировке ему придётся искать подстроку в строке, а он до сих пор не знает, как это делается. Помогите ему!

Формат входных данных

В первых двух строчках даны две строки $\mathcal{T}$ и $\mathcal{S}$ ($1 \leqslant |\mathcal{T}| \leqslant 1\,000\,000$, $1 \leqslant |\mathcal{S}| \leqslant 1\,000\,000$). Строки состоят только из маленьких букв «a»–«z» английского алфавита.

Формат выходных данных

Выведите через пробел все начала вхождений строки $\mathcal{S}$ в строку $\mathcal{T}$ в порядке возрастания. Вхождение начинается в позиции k, если $\mathcal{T}_k = \mathcal{S}_1, \mathcal{T}_{k+1} = \mathcal{S}_2, \ldots, \mathcal{T}_{k+|\mathcal{S}|-1} = \mathcal{S}_{|\mathcal{S}|}$. Если вхождений нет, выведите единственное слово «none» вместо списка вхождений.

Примеры

стандартный ввод	*стандартный вывод*
ababcbc abc	3
qwerty asdfgh	none

Задача B. Сравнение строк

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Циклическое расширение $\mathcal{S}^*$ строки $\mathcal{S}$ — это строка $\mathcal{S}$, приписанная сама к себе бесконечное количество раз; к примеру, циклическим расширением строки «bab» является строка «babbabbab...».

Даны длины двух строк $\mathcal{S}$ и $\mathcal{T}$ — числа m и n. Рассмотрим циклические расширения $\mathcal{S}^*$ и $\mathcal{T}^*$ этих строк. Как проверить, равны ли они?

Наивный алгоритм будет проверять строки на равенство, просто сравнивая s_1 с t_1, s_2 с t_2 и так далее. В итоге алгоритм либо найдёт пару несовпадающих символов, либо, если строки равны, будет проводить сравнения бесконечно долго.

Однако понятно, что, если строки не равны, то последняя позиция p, на которой мы можем встретить различие ($s_p \neq t_p$), конечна. Мы хотим узнать, чему равно p, чтобы улучшить алгоритм сравнения так: новый алгоритм будет сравнивать символ s_1 с t_1, s_2 с t_2 и так далее, пока либо не найдёт несовпадение $s_q \neq t_q$ на позиции $q \leqslant p$, либо, сравнив первые p пар и обнаружив соответствие символов в каждой паре, не докажет тем самым равенство строк $\mathcal{S}^*$ и $\mathcal{T}^*$.

Для данных длин строк m и n приведите пример строк $\mathcal{S}$ и $\mathcal{T}$ соответствующей длины, циклические расширения которых различны, но первое несовпадение встречается как можно позже.

Формат входных данных

В первой строке ввода заданы два числа через пробел — это числа m и n ($1 \leqslant m, n \leqslant 100$).

Формат выходных данных

Выведите в первой строке строку $\mathcal{S}$ из m символов, а во второй — строку $\mathcal{T}$ из n символов. Строки могут содержать только маленькие буквы английского алфавита ('a'–'z').

Пример

стандартный ввод	*стандартный вывод*
2 4	aa aaab

Пояснение к примеру

Для $m = 2$ и $n = 4$ значение p равно четырём, и оно достигается на строках «aa» и «aaab», циклические расширения которых — строки «aaaa...» и «aaab...» — различаются в четвёртом символе.

Задача C. Сравнения подстрок

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Дана строка. Нужно уметь отвечать на запросы следующего вида: равны ли подстроки $[a..b]$ и $[c..d]$.

Формат входных данных

Сперва строка S (не более 10^5 строчных латинских букв). Далее число M — количество запросов.

В следующих M строках запросы, каждый — в виде четырёх целых чисел a, b, c, d ($0 \leqslant m \leqslant 10^5$, $1 \leqslant a \leqslant b \leqslant |S|$, $1 \leqslant c \leqslant d \leqslant |S|$).

Формат выходных данных

Выведите M строк: на каждый запрос выведите «`Yes`», если подстроки совпадают, и «`No`» иначе.

Пример

стандартный ввод	*стандартный вывод*
trololo 3 1 7 1 7 3 5 5 7 1 1 1 5	Yes Yes No

Задача D. Взлом хеширования

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Ваши решения не работают на крайних случаях?
Встроенная быстрая сортировка неожиданно стала работать за квадратичное время?
В геометрических задачах не хватает точности вычислений?
Решение проходит локальное стресс-тестирование, но не работает на тестах жюри?
Именно в вашем случае ошибка оказалась не в решении, а в библиотечной функции?

Хотите узнать, кто за всем этим стоит?

Сегодня у вас есть уникальная возможность вступить в тайную организацию:
Орден Коварных Бобров! Члены этой организации делают в среднем на 146% больше успешных взломов, чем непосвящённые, а в задачи их авторства тесты приходится добавлять в несколько раз реже. Чтобы подать заявку на вступление, необходимо пройти вступительное испытание: решить предложенную ниже задачу.

Торопитесь! Количество мест ограничено!

В этой задаче требуется найти коллизию при полиномиальном хешировании строк, состоящих из маленьких букв английского алфавита.

Полиномиальный хеш строки имеет два параметра: множитель p и модуль q. Для пустой строки ε значение хеш-функции $h(\varepsilon) = 0$, а для любой строки S и любого символа c хеш-функция рекуррентно определяется как $h(S + c) = (h(S) \cdot p + \text{code}(c)) \bmod q$. Здесь $\text{code}(c)$ — это ASCII-код символа c. Как известно, коды маленьких букв английского алфавита идут подряд: $\text{code}(\texttt{'a'}) = 97$, $\text{code}(\texttt{'b'}) = 98$, ..., $\text{code}(\texttt{'z'}) = 122$. Можно выписать и нерекуррентную формулу: если строка $S = s_1 s_2 \dots s_n$, то $h(S) = (\text{code}(s_1) \cdot p^{n-1} + \text{code}(s_2) \cdot p^{n-2} + \dots + \text{code}(s_n) \cdot p^0) \bmod q$.

По заданным числам p и q найдите две различные непустые строки A и B такие, что $h(A) = h(B)$.

Формат входных данных

Первая строка ввода содержит два целых числа p и q, разделённых пробелом — параметры функции хеширования ($0 < p < q < 2 \cdot 10^{18}$).

Формат выходных данных

В первых двух строках выведите две различные непустые строки A и B, для которых $h(A) = h(B)$. Строки должны состоять исключительно из маленьких букв английского алфавита (ASCII-коды 97–122) и иметь длину от 1 до 100 000 символов. Заметим, что длины строк не обязательно должны совпадать. Если возможных ответов несколько, разрешается вывести любой из них.

Пример

стандартный ввод	*стандартный вывод*
31 47	aa bq

Пояснение к примере

В примере $h(A) = (97 \cdot 31 + 97) \bmod 47 = 3104 \bmod 47 = 2$ и $h(B) = (98 \cdot 31 + 113) \bmod 47 = 3151 \bmod 47 = 2$.

Задача E. Период строки

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Строка S имеет период T, если

$$\exists n > 0 : S = T^n = \underbrace{TT\ldots T}_{n}.$$

Вам дана строка S. Ваша задача — найти минимальную по длине строку T, для которой $S = T^n$ при некотором $n \in \mathbb{N}$.

Формат входных данных

Строка S длиной от 1 до 10^6 символов.

Формат выходных данных

Единственное число — длина T.

Пример

стандартный ввод	*стандартный вывод*
abaabaabaabaaba	3

Задача F. Два генератора

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Строка p называется *генератором* строки s, если s совпадает с каким-то префиксом p^*, и p — кратчайшая из таких строк. Здесь p^* — это строка p, приписанная к себе самой бесконечное количество раз.

Рассмотрим строку t. Представим её в виде конкатенации $t = a+b$ (строки a и b могут быть пустыми). После этого найдём генератор строки a и генератор строки b. Найдите такое разбиение t на a и b, что суммарная длина этих двух генераторов минимальна, и выведите эту длину.

Формат входных данных

В первой строке задано целое число n — длина строки t ($1 \leqslant n \leqslant 300\,000$). Во второй строке записана сама строка t, состоящая из строчных букв английского алфавита.

Формат выходных данных

Выведите одно целое число: минимально возможную суммарную длину двух генераторов.

Примеры

стандартный ввод	*стандартный вывод*
10 abcabcaddd	4
3 aaa	1

Задача G. От префикс-функции к z-функции

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Префикс-функция $p(i)$ для строки $s = s_1 s_2 \dots s_n$ определяется от позиции i ($1 \leqslant i \leqslant n$) в строке так: $p(i)$ — это максимальная длина собственного префикса строки $s_1 s_2 \dots s_i$, равного её собственному суффиксу. Напомним, что *собственный префикс* строки $s = s_1 s_2 \dots s_n$ — это строка $s_1 s_2 \dots s_r$ для некоторого $r < n$. Аналогично, *собственный суффикс* строки $s = s_1 s_2 \dots s_n$ — это строка $s_l s_2 \dots s_n$ для некоторого $l > 1$.

Z-функция $z(i)$ для строки $s = s_1 s_2 \dots s_n$ определяется от позиции i ($1 \leqslant i \leqslant n$) в строке так: $z(1) = 0$, а для $i > 1$ значение $z(i)$ — это максимальное число, для которого строки $s_1 s_2 \dots s_{z(i)}$ и $s_i s_{i+1} \dots s_{i+z(i)-1}$ совпадают.

Даны длина строки n и значения префикс-функции $p(1)$, $p(2)$, …, $p(n)$ для этой строки. Найдите для этой строки значения z-функции $z(1)$, $z(2)$, …, $z(n)$.

Формат входных данных

В первой строчке задано целое число n ($1 \leqslant n \leqslant 1\,000\,000$). Во второй строчке заданы n чисел через пробел — значения префикс-функции $p(1)$, $p(2)$, …, $p(n)$. Гарантируется, что существует строка длины n, состоящая из маленьких букв английского алфавита, для которой префикс-функция от позиций 1, 2, …, n принимает данные значения.

Формат выходных данных

В первой строчке выведите n чисел через пробел — значения z-функции для строки, имеющей данную префикс-функцию.

Примеры

стандартный ввод	*стандартный вывод*
6 0 0 1 2 3 4	0 0 4 0 2 0
7 0 0 0 1 2 3 4	0 0 0 4 0 0 1
4 0 0 0 0	0 0 0 0

Задача H. От z-функции к префикс-функции

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Z-функция $z(i)$ для строки $s = s_1 s_2 \dots s_n$ определяется от позиции i ($1 \leqslant i \leqslant n$) в строке так: $z(1) = 0$, а для $i > 1$ значение $z(i)$ — это максимальное число, для которого строки $s_1 s_2 \dots s_{z(i)}$ и $s_i s_{i+1} \dots s_{i+z(i)-1}$ совпадают.

Префикс-функция $p(i)$ для строки $s = s_1 s_2 \dots s_n$ определяется от позиции i ($1 \leqslant i \leqslant n$) в строке так: $p(i)$ — это максимальная длина собственного префикса строки $s_1 s_2 \dots s_i$, равного её собственному суффиксу. Напомним, что *собственный префикс* строки $s = s_1 s_2 \dots s_n$ — это строка $s_1 s_2 \dots s_r$ для некоторого $r < n$. Аналогично, *собственный суффикс* строки $s = s_1 s_2 \dots s_n$ — это строка $s_l s_2 \dots s_n$ для некоторого $l > 1$.

Даны длина строки n и значения z-функции $z(1)$, $z(2)$, …, $z(n)$ для этой строки. Найдите для этой строки значения префикс-функции $p(1)$, $p(2)$, …, $p(n)$.

Формат входных данных

В первой строчке задано целое число n ($1 \leqslant n \leqslant 1\,000\,000$). Во второй строчке заданы n чисел через пробел — значения z-функции $z(1)$, $z(2)$, …, $z(n)$. Гарантируется, что существует строка длины n, состоящая из маленьких букв английского алфавита, для которой z-функция от позиций 1, 2, …, n принимает данные значения.

Формат выходных данных

В первой строчке выведите n чисел через пробел — значения префикс-функции для строки, имеющей данную z-функцию.

Примеры

стандартный ввод	*стандартный вывод*
6 0 0 4 0 2 0	0 0 1 2 3 4
7 0 0 0 4 0 0 1	0 0 0 1 2 3 4
4 0 0 0 0	0 0 0 0

Задача I. Минимальный суффикс

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Как известно, *k-м суффиксом* строки $S = s_1 s_2 \dots s_n$ называется строка $S_k = s_k s_{k+1} s_{k+2} \dots s_n$.

Например, для строки $S =$ `abaca` суффиксы будут такие: $S_1 =$ `abaca`, $S_2 =$ `baca`, $S_3 =$ `aca`, $S_4 =$ `ca`, $S_5 =$ `a`, а все последующие суффиксы пусты.

В этой задаче требуется найти лексикографически минимальный непустой суффикс заданной строки S.

Формат входных данных

В первой строке записана строка S, состоящая только из маленьких букв английского алфавита. Длина этой строки — от 1 до 100 000 букв, включительно.

Формат выходных данных

В первой строке выведите суффикс S_k, являющийся лексикографически минимальным непустым суффиксом строки S.

Примеры

стандартный ввод	*стандартный вывод*
abcde	abcde
abaca	a
bcdeabc	abc

Задача J. Палиндромы

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мебибайт

Строка называется палиндромом, если она одинаково читается как слева направо, так и справа налево. Например, `abba` — палиндром, а `omax` — нет. Для строки S будем обозначать $S[i..j]$ её подстроку длины $j - i + 1$ с i-й по j-ю позицию включительно (позиции нумеруются с единицы).

Для заданной строки S длины N требуется найти количество таких пар (i, j), что $1 \leqslant i < j \leqslant n$ и подстрока $S[i..j]$ является палиндромом.

Формат входных данных

Ввод содержит одну строку S длины N, состоящую из маленьких английских букв ($1 \leqslant N \leqslant 100\,000$).

Формат выходных данных

Выведите искомое количество пар.

Примеры

стандартный ввод	*стандартный вывод*
aaa	3
abba	2
omax	0