Prof. Жужжащий профессор

Имя входного файла: `prof.in`
Имя выходного файла: `prof.out`

В одном очень известном университете один очень известный профессор очень быстро произносил свои лекции, так, что ничего невозможно было разобрать. Студенты шутили по этому поводу, что он не говорит, а жужжит. Естественно, что про загадочного профессора никто абсолютно ничего не знал.

Но вот недавно Петя Булочкин решил предпринять исследование по изучению словарного запаса профессора. С этой целью он даже посетил одну лекцию и записал все сказанное на ней на диктофон. Затем, прокручивая дома запись с десятикратным замедлением, Петя смог записать все, что сказал профессор. Но вот незадача — профессор говорил так быстро, что, даже прослушивая замедленную запись, нельзя было точно сказать, где он делал паузы между словами. Таким образом, у Пети есть некоторый текст S, состоящий только из маленьких латинских букв — лекция, которая была прочитана профессором.

Петя решил, что те слова, которые профессор употреблял только один раз во время своей лекции, его не интересуют. Кроме того, понятно, что если профессор употреблял некоторое слово два или более раз, то существуют два неперекрывающихся вхождения этого слова в текст S. Назовем непустую строку T кандидатом в слова, если существуют два неперекрывающихся вхождения T в S. Теперь Петя хочет найти все строки, которые являются кандидатами в слова. И поможете ему в этом Вы.

Формат входного файла

Единственная строка входного файла содержит от 1 до 3000 маленьких латинских букв. Это и есть текст S, который прочитал профессор на лекции.

Формат выходного файла

Единственная строка выходного файла должна содержать одно число, равное количеству строк, являющихся кандидатами в слова.

Пример

prof.in	prof.out
bbaabbbabb	7

Cubes. Кубики

Имя входного файла: `cubes.in`
Имя выходного файла: `cubes.out`

Привидение Петя любит играть со своими кубиками. Он любит выкладывать их в ряд и разглядывать свое творение. Однако недавно друзья решили подшутить над Петей и поставили в его игровой комнате зеркало. Ведь всем известно, что привидения не отражаются в зеркале! А кубики отражаются.

Теперь Петя видит перед собой N цветных кубиков, но не знает, какие из этих кубиков настоящие, а какие — всего лишь отражение в зеркале. Помогите Пете! Выясните, сколько кубиков может быть у Пети. Петя видит отражение всех кубиков в зеркале и часть кубиков, которая находится перед ним. Часть кубиков может быть позади Пети, их он не видит.

Формат входного файла

Первая строка входного файла содержит число N ($1 \leqslant N \leqslant 100\,000$) и количество различных цветов, в которые могут быть раскрашены кубики — M ($1 \leqslant M \leqslant 100\,000$). Следующая строка содержит N целых чисел от 1 до M — цвета кубиков.

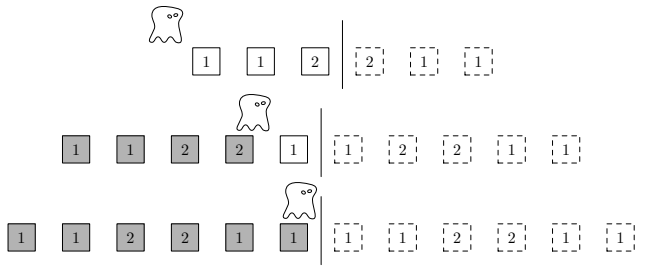
Формат выходного файла

Выведите в выходной файл все такие K, что у Пети может быть K кубиков.

Пример

cubes.in	cubes.out
6 2 1 1 2 2 1 1	3 5 6

В приведенном примере взаимные расположения Пети, кубиков и зеркала приведены на рисунке. Петя смотрит вправо, затененные на рисунке кубики находятся позади Пети, и поэтому он их не видит.



Воссановление HTML файла

Имя входного файла: `html.in`
Имя выходного файла: `html.out`

HTML

Пея недавно скачал поврежденный HTML файл. Он выглядит как обычный HTML файл, но в нем есь некоторые несоответствия тегов. Ваша задача — удалив минимальное количесво открывающихся и закрывающихся эгов сделать так что бы структура тегов стала правильной.

Более формально, HTML файл сосоит из произвольных символов с ASCII кодами из диапазона о 32 до 126, а также Linux-style переводов строки (символов с кодом 10). Тэги окрываются следующим образом: *<Имя эга Параметры>* и закрываются следующим образом *</Имя эга>*. Имя тега — строка, сосоящая из больших и маленьких латинских букв, которые считаются различными. *Имя эга* отделяется от *Параметров* как минимум

одним пробелом (но не переводом сроки). Параметры могут содержать произвольные допустимые ASCII символы кроме <, > и переводов срок. Также допускаются открывающиеся теги без парамеров и записываются в следующей форме: <*Имя Тэга*>.

HTML файл счиается правильным, если каждому открывающемуся тегу можно привеси в соответствие следующий далее в файле закрывающийся тег таким образом чобы часть файла между этими тегами также представляла собой правильный HTML файл, и аналогично можно привеси в соответствие каждому закрывающемуся тегу ровно один окрывающийся тег идущий ранее в файле. HTML файл не содержащий эгов также является правильным.

Окрывающийся тег считается соответствующим закрывающемуся если у них одинаковое имя эга.

Формат входного файла

Входной файл предсавляется из себя поврежденный HTML файл, коорый требуется исправить. Его длина не превосходит 10000 байт. Количесво открывающихся и закрывающихся тегов не превышает 500. Символы < и > не всречаются нигде кроме окрывающихся и закрывающихся тегов.

Формат выходного файла

Выведие единственное целое число — минимальное количество окрывающихся/закрывающихся тегов, которые требуется удалить, чтобы файл оказался правильным HTML файлом.

Пример

html.in
 <b someone has corrupted this file> It was a good file before...
html.out
2

html.in
<a>That's good
html.out
0

Цензура

Имя входного файла: `censored.in`
Имя выходного файла: `censored.out`

Посчитайте, сколько строк над алфавитом из n символов длины m не содержат ни одной подстроки из заданного множества “запрещенных” строк.

Формат входного файла

В первой строке написаны целые числа n ($1 \leqslant n \leqslant 100$) — количество символов в алфавите, m ($1 \leqslant m \leqslant 100$) — длина искомых строк и p ($0 \leqslant p \leqslant 10$) — количество “запрещенных” подстрок. Следующая строка содержит n символов с кодами больше 32 — буквы алфавита. Далее идет p “запрещенных” строк, длины которых не превосходят $\min(m, 10)$ символов. Строки целиком состоят из символов алфавита.

Формат выходного файла

В первой строке выведите ответ на задачу.

Пример

censored.in	censored.out
2 3 1 AB AA	5
2 100 4 AB AA AB BA BB	0