Задача A. Строковый компьютер

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

Василий Афанасьев в качестве курсовой работы получил задание — построить компьютер, котрый бы работал не с числами, а со строками. Вася для начала фиксировал некоторый алфавит. Обозначим за K количество букв в этом алфавите. Далее, Вася фиксировал некоторый набор различных строк, длины не более N каждая, который он назвал базовым. Компьютер умеет работать только со строками, которые получаются конкатенацией (т.е. приписыванием) некоторых строк из этого набора друг к другу (одну и ту же строку при приписывании можно использовать несколько раз). Однако оказалось, что исходный базовый набор оказался *чрезмерным*! Это значит, что в нем была строка, при удалении которой из набора не изменяется множество строк, с которыми умеет работать компьютер.

Васю заинтересовал вопрос — как много может быть строк в нечрезмерном наборе, и сколько таких максимальных наборов существует.

Так как его компьютер еще не готов, то он попросил Вас посчитать это число.

Формат входного файла

Во входном файле содержится два целых числа $1 \le N \le 1\,000$ и $1 \le K \le 100$ — соответственно максимальная длина строки и количество символов в алфавите.

Формат выходного файла

На первой строке выведите количество чисел в максимальном наборе. Во второй — количество таких наборов.

Пример

computer.in	computer.out
1 1	1 1
3 2	12 1

Задача B. Циклическое k-расширение

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

Вася недавно узнал, что такое циклическое k-расширение строки S. Его можно получить следующим образом: склеить k экземпляров строки S, а потом взять первые k символов результата.

Узнав это, Вася обрадовался, взял некоторую строку, и начал к ней применять описанную операцию, не запоминая, какое он каждый раз брал k.

Вам дана часть строки, получившейся у Васи. Ваша задача определить, не ошибся ли Вася в своих сложных преобразованиях, т. е., мог ли у него из первоначальной строки получиться ответ, содержащий данную строку в качестве подстроки.

Формат входного файла

В первой строке входного файла находится изначальная строка, которую Вася бережно записал перед тем, как приступить к своим действиям. Во второй строке находится подстрока результата, полученного Васей. Обе строки непусты и по длине не превышают $5\,000$ символов. Строки могут состоять из больших и маленьких латинских букв (с учетом регистра), а также цифр.

Формат выходного файла

Выведите “`NO`”, если можно точно сказать, что Вася ошибся, и “`YES`”, если мог и не ошибиться.

Пример

cycle.in	**cycle.out**
`abc` `abc`	`YES`
`abcd` `bcabc`	`YES`
`abcabc` `abcA`	`NO`

Задача C. Удаление

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

Вася уже долго занимается k-расширениями. Недавно ему это надоело, и он стал изучать следующую операцию на строках: сначала удаляется каждый k-ый символ строки с начала и до конца (т. е. сначала k-й, потом $2k$-й, и т. д., пока число ik не превосходит длины строки). Потом эта операция повторяется, и так далее, пока в строке есть хотя бы k символов.

Например, если дана строка длины 10, и $K = 2$, то сначала будут удалены 2, 4, 6, 8 и 10 символы. Затем будут удалены 2 и 4 символы новой строки, которые соответствуют 3 и 7 символам старой. Затем будет удален 2 символ получившейся строки — 5 символ старой. Последним будет удален 9 символ старой строки.

Собственно, задумал он это все для того, чтобы узнать, какой символ когда исчезнет. Но когда ему надоело вручную удалять символы, он решил поручить это Вам. А именно, Вам придется отвечать на его вопросы: "А на какой секунде был удален i-ый символ строки?" На удаление каждого символа тратится одна секунда.

Формат входного файла

В первой строке находятся три числа: $1 \le n \le 5 \cdot 10^6$ — количество символов, которые написал Вася, число $1 \le k \le n$, и $1 \le l \le 10\,000$ — количество вопросов, которые задает Вася. В последующих l строках указаны номера символов, для которых Вася хочет узнать, когда они были удалены. Все номера лежат в пределах от 1 до n и могут повторяться.

Формат выходного файла

Выведите в выходной файл l строк с ответами на вопросы в том порядке, в каком они были во входном файле. Если окажется, что символ из строки не будет удален никогда, выведите 0.

Пример

deletion.in	deletion.out
10 2 6 1 2 3 5 10 5	0 1 6 8 5 8

Задача D. ДНК

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

Вася никогда не любил биологию. Но когда он узнал про ДНК, у него появился живой интерес. Он решил, что если все существа произошли друг от друга, то и ДНК у них должны быть похожими. У некоторых более похожие, у некоторых — менее, но у всех ДНК можно записать в виде строки, состоящей из символов A, C, G и T. Поэтому он решил найти какой-нибудь показатель родства. И придумал следующее. Он берет из двух ДНК по подстроке. Если одна из них является анаграммой другой (т. е. получается перестановкой букв), то это хорошая пара подстрок. Естественно, в любой хорошей паре обе подстроки имеют одинаковую длину. Тогда степень родства двух ДНК — это максимально возможная длина подстрок в хорошей паре.

Формат входного файла

В первой строке находится ДНК Васи. А во второй строке — ДНК первого попавшегося Васе живого существа. Обе строки непусты и состоят не более, чем из 1 300 символов A, C, G и T.

Формат выходного файла

В первую строку выходного файла выведите степень родства Васи с подопытным существом.

Если степень родства отлична от нуля, то во вторую следует вывести две начальные позиции подстрок из соответствующей хорошей пары в первой и второй ДНК соответственно. В случае неоднозначности последних двух чисел, выведите любые подходящие.

Пример

dna.in	dna.out
ACGT GTC	3 2 1
ACGT CAT	2 1 1
ACA AC	2 2 1

Задача E. Слова

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

Вася умеет быстро набирать на клавиатуре разные слова. Иногда он делает это так быстро, что в слове меняются местами какие-то две буквы (не обязательно стоящие подряд). И тогда, если Вася общается в чате, собеседник не всегда понимает его правильно: ведь, скажем, если Вася набрал «BEATS» и при этом, возможно, поменял местами две буквы, он мог иметь в виду и «BEATS», и «BEAST», и даже «BETAS»...

Как хорошо бы было, если бы осмысленные слова нельзя было перепутать, даже если переставить в них какие-то две буквы! Васю заинтересовала теоретическая сторона этого вопроса. А именно: сколько же можно выделить слов из заданного набора букв так, чтобы никакие два слова, если в одном из них или даже в каждом переставить две буквы местами, не стали бы одинаковыми. Например, множество слов «BEAST» и «BETAS» не подходит, потому как из каждого слова можно перестановкой двух букв получить «BEATS». С другой стороны, «WORDS» и «SWORD» — подходящее множество: как ни переставляй пару букв в одном и в другом слове, одинаковую последовательность не получить.

Вася хочет по набору букв выяснить, какое максимальное количество слов, являющихся перестановками букв этого набора, можно объявить осмысленными, чтобы никакие два из них нельзя было перепутать, переставив пару букв в одном или в обоих словах. Помогите ему справиться с этой задачей.

Формат входного файла

В первой строке входного файла записаны подряд пять заглавных букв латинского алфавита.

Формат выходного файла

Выведите в выходной файл максимальное по размеру множество слов, являющихся перестановками данного набора букв, которые можно объявить осмысленными, чтобы их нельзя было перепутать. Слова следует выводить по одному на строке. Если таких множеств несколько, разрешается вывести любое из них.

Пример

words.in	**words.out**
TATRA	TATRA ARATT

Задача F. Дана строка!!!

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

Васе уже надоели задачи на строки! А Вам? А что делать? Что ж, приступим. Дана строка из маленьких букв латинского алфавита. Разрешается любой ее символ сдвинуть не более, чем на K позиций в любую сторону так, чтобы в конечном счете они все встали на разные позиции. Например, если строка — aababac, а $K = 2$, то таким образом можно получить строки abaaabc или aaaabbc, но нельзя — aaacbab или aaaacbb.

Вася хочет сделать так, чтобы получившаяся с помощью такой операции строка была минимально возможной лексикографически (т.е. расположена раньше всех по правилам упорядочивания слов в словаре). Как же ему быть?

Формат входного файла

В первой строке входного файла задано число K ($K \geq 0$). Во второй строке задана сама исходная непустая строка, длиной не более 100 000 маленьких латинских букв. Гарантируется, что K не превосходит длины строки.

Формат выходного файла

Выведите лексикографически минимальный из возможных результатов.

Пример

string.in	string.out
2 aababac	aaaabbc

Задача G. Ключи

Имя входного файла:	`keys.in`
Имя выходного файла:	`keys.out`
Ограничение по времени:	5 секунд
Ограничение по памяти:	64 мегабайта

Для доступа в лаборатории НИИ Исследований Данных Строк используются ключи в виде прямоугольных карточек $N \times M$, в которых вырезаны дырки. Эти ключи можно вставлять только одним способом (то есть ни поворачивать, ни переворачивать нельзя). При этом дырки имеют прямоугольную форму. К Васе попало два ключа от разных лабораторий. Он решил их наложить друг на друга так, чтобы получившаяся фигура имела максимальное количество дырок (просветов). При этом исходно ключи лежали в том положении, в котором их необходимо вставлять в замок, а Вася не хочет их поворачивать. Помогите Васе определить максимальное число дырок. При наложении считаются только те дырки, внутренности которых непусты.

Формат входного файла

Первая строка содержит два целых числа — $1 \leq N, M \leq 10^9$ — длины сторон ключа. Вторая строка содержит единственное целое число — $1 \leq K_1 \leq 500$ — число дырок в первом ключе. Далее в K_1 строках написано по четыре целых числа — $X_1\ Y_1\ X_2\ Y_2$ $(0 \leq X_1 < X_2 \leq N, 0 \leq Y_1 < Y_2 \leq M)$ — координаты углов соответствующих прямоугольных дырок. Дырки в ключе не пересекаются и не касаются.

Далее следует описание второго ключа в таком же формате.

Формат выходного файла

Выведите единственное целое число — максимальное количество дырок, которое может получить Вася.

Пример

keys.in	keys.out
10 10 1 1 1 2 2 1 1 1 2 2	1
10 10 2 1 1 2 2 3 3 4 4 1 1 1 2 2	1
10 10 2 1 1 2 2 3 3 4 4 1 1 1 3 3	2

Задача H. Замечательные дороги

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

В одной замечательной стране живут замечательные люди. По исследованиям замечательного правительства, большинство граждан на выходных садятся в машину, выбирают циклический маршрут между некоторыми городами и деревнями без повторяющихся населенных пунктов и катаются по этому маршруту, пока не надоест. Некоторые, правда, катаются по своему городу и никуда не выезжают.

Так как правительство заботится о своих гражданах, оно хочет сделать их выходные максимально красочными. По этой причине недавно было принято решение покрасить каждую дорогу между населенными пунктами в какой-нибудь цвет. Причем так, чтобы ни на каком "выходном" маршруте не было дорог одинакового цвета. Но так как цветов могло понадобиться довольно много, правительство решило минимизировать количество различных цветов. Вам предстоит помочь этому замечательному государству в осуществлении его планов.

Формат входного файла

В первой строке находятся два числа: $1 \le n \le 50\,000$ — количество городов и деревень в стране и $1 \le m \le 100\,000$ — количество дорог. В m последующих строках находится по два числа — номера населенных пунктов, концов дороги. Ни одна дорога не ведет из города в себя, и между двумя населенными пунктами не может быть более одной дороги. Все дороги двусторонние.

Формат выходного файла

В первой строке выведите минимальное количество цветов. В последующих m строках выведите по три числа: два конца дороги в любом порядке и ее цвет. Дороги разрешается выводить в произвольном порядке.

Пример

roads.in	**roads.out**
2 1 1 2	1 2 1 1
3 2 1 2 2 3	1 2 3 1 1 2 1
3 3 1 2 2 3 3 1	3 1 2 1 3 2 2 3 1 3