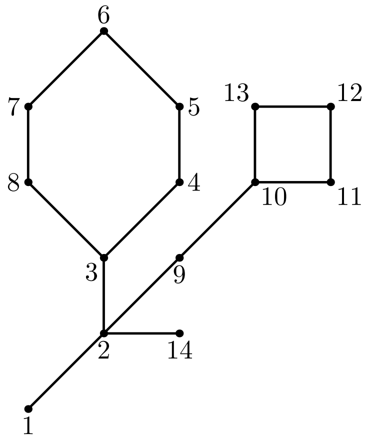
Задача A. Кактусы

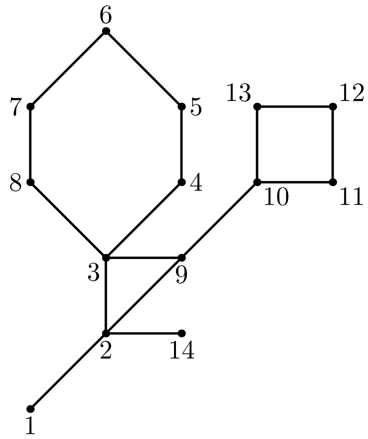
Имя входного файла:	`cacti.in`
Имя выходного файла:	`cacti.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Вершинный кактус — это связный неориентированный граф, в котором каждая вершина лежит не более чем на одном простом цикле.

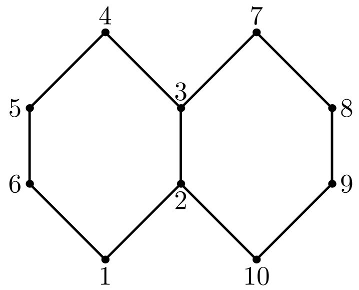
В свою очередь, *реберный кактус* — это связный неориентированный граф, в котором каждое ребро лежит не более чем на одном цикле. Отметим, что любой вершинный кактус является реберным, но не наоборот, пример реберного кактуса, который не является вершинным, приведен на рисунке.



Вершинный кактус



Реберный кактус, который не является вершинным



Не кактус

Не кактус

По заданному n найдите количество помеченных вершинных и реберных кактусов с n вершинами. Выведите эти числа по модулю $10^9 + 7$.

Формат входного файла

Входной файл содержит n ($1 \le n \le 50$).

Формат выходного файла

Выходной файл должен содержать количество помеченных вершинных и количество помеченных реберных кактусов с n вершинами, оба числа должны быть выведены по модулю $10^9 + 7$.

Примеры

`cacti.in`	`cacti.out`
2	1 1
5	347 362

Задача B. Дигитация

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

В некоторых задачах встречается такая функция от целого неотрицательного числа, как сумма цифр. Например, установить, делится ли число на 9, достаточно просто: нужно лишь определить, делится ли сумма цифр числа на 9.

Рассмотрим обобщение суммы цифр числа — *дигитацию*. А именно, если сумма цифр числа $s(x)$ состоит из одной цифры, то дигитация $d(x)$ равна $s(x)$. Иначе, применяем $s(x)$ к сумме цифр, пока результат не будет состоять из одной цифры. Это можно выразить и короче: $d(x) = d(s(x))$.

Задано число n. Необходимо вычислить дигитацию факториала этого числа: $d(n!) = d(1 \cdot 2 \cdot \ldots \cdot n)$.

Формат входного файла

В первой строке входного файла содержится единственное число n ($1 \le n \le 10^9$).

Формат выходного файла

В выходной файл выведите дигитацию $n!$.

Примеры

digits.in	digits.out
2	2
3	6

Задача C. Зайцы в клетках

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

Всем известен, так называемый, принцип Дирихле, который формулируется следующим образом.

Предположим, что некоторое число кроликов рассажены в клетках. Если число кроликов больше, чем число клеток, то хотя бы в одной из клеток будет больше одного кролика.

В данной задаче мы рассмотрим более общий случай этого классического математического факта. Пусть есть N клеток и M зайцев, которых рассадили по этим клеткам. Вам требуется рассчитать максимальное количество зайцев, которое гарантированно окажется в одной клетке.

Формат входного файла

В первой строке входного файла записаны два натуральных числа N и M ($1 \le N, M \le 10^9$).

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

hares.in	hares.out
2 3	2

Задача D. Платные дороги

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

Скорому открытию ЗСД посвящается.

Мэр одного большого города решил ввести плату за проезд по шоссе, проходящим в районе города, чтобы снизить объем транзитного транспорта. В районе города проходит n шоссе.

Но руководство области, в которой расположен город, воспротивилось планам мэра. Действительно — дальнобойщики представляют собой неплохой источник доходов для большого количества кафе и гостиниц в небольших городках.

В результате решили, что плата будет введена только на шоссе, которые *проходят через город*.

В городе используется развитая система метрополитена, всего в городе есть m станций метро. Решено было, что шоссе проходит через город, если либо одна из станций метро расположена непосредственно на шоссе, либо есть хотя бы одна станция с каждой стороны от шоссе.

Помогите теперь мэру определить, какие шоссе проходят через город.

Формат входного файла

Первая строка входного файла содержит два целых числа: n и m — количество шоссе и количество станций метро, соответственно ($1 \le n, m \le 100\,000$).

Следующие n строк описывают шоссе. Каждое шоссе описывается тремя целыми числами a, b и c и представляет собой прямую на плоскости, задаваемую уравнением $ax+by+c=0$ ($|a|, |b|, |c| \le 10^6$).

Следующие m строк входного файла описывают станции метро. Каждая станция описывается двумя целыми числами x и y и представляет собой точку на плоскости с координатами (x, y) ($|x|, |y| \le 10^6$).

Формат выходного файла

Первая строка выходного файла должна содержать одно целое число — количество шоссе, которые проходят через город. Вторая строка должна содержать номера этих шоссе в возрастающем порядке. Шоссе нумеруются от 1 до n в порядке, в котором они описаны во входном файле.

Пример

highways.in	highways.out
4 2 0 1 0 1 0 1 1 1 0 1 1 -1 0 0 2 0	3 1 3 4

Задача E. Ленивый студент

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

Вася — гениальный программист. По крайней мере, он так считает. Поэтому все, что рассказывают на уроках информатики, ему давно известно. Поэтому вместо того, чтобы слушать преподавателя, Вася придумал следующую игру: он берет с собой газету и вычеркивает в тексте все буквы, содержащие «полости». Например, он вычеркивает буквы o и a, но пропускает w и c. Вася считает, что таким образом он тренирует внимательность. Однако ему как-то надо проверять, что он нигде не ошибся. Он считает, что для этого достаточно проверить, что количество вычеркнутых букв верно. Поэтому он просит вас написать программу, определяющую, сколько букв должно быть вычеркнуто в данном тексте.

Формат входного файла

В первой строке входного файла задана строка из строчных латинских букв и пробелов — текст, который Вася брал с собой на урок. Длина строки не превышает 100000 символов. Газета, из которой Вася взял этот текст использует те же шрифты, что и данное условие.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

lazy.in	lazy.out
`a sample sentence`	`7`
`jing jang walla walla bing bang`	`12`
`abcdefghijklmnopqrstuvwxyz`	`8`

Задача F. Разбиения

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

Разбиением множества $S = \{1, 2, \ldots, n\}$ называется набор $\mathcal{P}$ множеств $\mathcal{P} = \{P_1, P_2, \ldots, P_k\}$, такой что $\bigcup P_i = S$ и $P_i \cap P_j = \varnothing$ для $i \neq j$. Примером разбиения для $n = 5$ может служить $P_1 = \{1, 3\}$, $P_2 = \{2, 4, 5\}$.

Говорят, что разбиение избегает множества Q если ни одно из множеств P_i не содержит Q в качестве подмножества. Например, разбиение, приведенное выше, избегает множеств $\{1, 2\}$ и $\{3, 4\}$, но не избегает, например, множеств $\{1, 3\}$ или $\{2\}$.

По заданному n и набору множеств $Q_1, Q_2, \ldots, Q_l$ найдите количество разбиений, которые избегают каждого из множеств Q_i.

Формат входного файла

Первая строка входного файла содержит два целых числа n и l ($1 \le n \le 100$, $0 \le l \le 10$). Следующие l строк описывают множества, которые следует избегать. Каждая строка начинается с целого числа q_i — размера множества, за которым следует q_i чисел — элементы множества.

Формат выходного файла

Выведите одно целое число — количество разбиений, которые избегают каждого из множеств Q_i.

Пример

partitions.in	partitions.out
5 2 3 1 2 3 2 2 4	34

Задача G. Строчечки

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

Строка $A = a_1 a_2 \dots a_n$ является *подстрокой* строки $B = b_1 b_2 \dots b_m$, если существует такое число k $(1 \le k \le m - n + 1)$, что $a_1 = b_k$, $a_2 = b_{k+1}$, $\dots$, $a_n = b_{k+n-1}$.

Строка B называется *надстрокой* A, если A является подстрокой B.

Строка $A = a_1 a_2 \dots a_n$ *лексикографически меньше* строки $B = b_1 b_2 \dots b_m$, если для некоторого k и для всех $1 \le t \le k$ верно $a_t = b_t$ и либо $a_{k+1} < b_{k+1}$, либо длина A равна k, а длина B больше k.

Даны S и T. Необходимо найти строку, являющуюся одновременно подстрокой S и надстрокой T. Если таких строк несколько, выведите самую длинную из них. Если строк наибольшей длины несколько, выведите лексикографически наименьшую.

Формат входного файла

В первой строке находится строка S $(1 \le |S| \le 4000)$. Во второй строке находится строка T $(1 \le |T| \le 4000)$. Обе строки состоят из строчных латинских букв. Здесь как $|S|$ и $|T|$ обозначены соответственно длины строк S и T.

Формат выходного файла

В выходной файл выведите искомую строку или «`NO SOLUTION`» (без кавычек), если такой строки не существует.

Примеры

strings.in	strings.out
bee be	bee
string nothing	NO SOLUTION

Задача H. Счастливый билетик — 2

Имя входного файла:	`ticket2.in`
Имя выходного файла:	`ticket2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Назовем *билетиком* последовательность цифр длины n. Билетик A называется *счастливым*, если существует число k ($1 \le k < n$) такое, что $\sum_{i=1}^{k} a_i = \sum_{i=k+1}^{n} a_i$. Число k при этом называется *границей счастья*.

Ваша задача — написать программу, определяющую для заданного билетика его наименьшую границу счастья, если она существует.

Формат входного файла

Первая строка входного файла содержит число n ($2 \le n \le 10^6$) — длина билетика A. Во второй строке содержатся цифры a_1, a_2, …, a_n ($0 \le a_i \le 9$), разделенные пробелами.

Формат выходного файла

Если билетик является счастливым, выведите его наименьшую границу счастья, в противном случае выведите «`-1`».

Примеры

ticket2.in	ticket2.out
4 3 2 1 6	3
4 1 2 3 4	-1

Задача I. XOR

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

Команда знаменитого криптоаналитика профессора Ксора работает над взломом нового Шифра Изощренно Защищенного от Анализа (ШИЗА). После долгих исследований удалось свести задачу взлома этого шифра к следующей.

Задано несколько целых чисел $a_1, a_2, \ldots, a_n$. Запишем их в двоичной системе счисления, добавив меньшим из них ведущих нулей таким образом, чтобы все они имели такое же количество двоичных цифр, как и максимальное из них. После этого требуется переупорядочить биты в них, получив новые числа $b_1, b_2, \ldots, b_n$. Необходимо сделать это таким образом, чтобы выполнялось $b_1 \oplus b_2 \oplus \ldots \oplus b_n = 0$. Здесь $\oplus$ означает побитовое исключающее «или» (`xor` в паскале, `^` в си и джаве).

Вам, как кандидату на включение в команду профессора, поручили разобраться с этой задачей.

Формат входного файла

Первая строка входного файла содержит n ($2 \le n \le 50$). Вторая строка содержит числа $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^{18}$).

Формат выходного файла

Выведите $b_1, b_2, \ldots, b_n$.

Если решения не существует, выведите «`impossible`».

Примеры

xor.in	xor.out
3 7 10 11	7 12 11
3 7 10 3	impossible

В первом примере $a_1 = 7 = 0111_2$, $a_2 = 10 = 1010_2$, $a_3 = 11 = 1011_2$. Если переупорядочить биты следующим образом: $b_1 = 7 = 0111_2$, $b_2 = 12 = 1100_2$, $b_3 = 11 = 1011_2$, получится $b_1 \oplus b_2 \oplus b_3 = 0$.