Задача A. Восходящие последовательности (версия для дивизиона 2)

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Рассмотрим последовательность $\langle a_1, a_2, \ldots, a_n \rangle$ из неотрицательных целых чисел. *Восхождением* в этой последовательности называется пара соседних элементов такая, что элемент с бо́льшим индексом имеет бо́льшее значение. Например, в последовательности $\langle 0, 2, 3, 1, 0 \rangle$ два восхождения: от $a_1 = 0$ к $a_2 = 2$, и от $a_2 = 2$ к $a_3 = 3$. Обозначим количество восхождений среди первых k элементов последовательности через A_k. В данном примере, $A_1 = 0$, $A_2 = 1$, $A_3 = 2$, $A_4 = 2$ и $A_5 = 2$.

Последовательность a называется *восходящей*, если $a_1 = 0$ и для любого $i \geqslant 2$ имеет место неравенство $a_i \leqslant A_{i-1} + 1$. Например, последовательность $\langle 0, 2, 3, 1, 0 \rangle$ не является восходящей, так как $a_2 = 2$ и $A_1 = 0$. Последовательность $\langle 0, 1, 0, 2, 3 \rangle$, в свою очередь, является восходящей, так как $A_1 = 0$, $A_2 = 1$, $A_3 = 1$, $A_4 = 2$.

Последовательность $\langle a_1, a_2, \ldots, a_n \rangle$ из неотрицательных целых чисел *избегает паттерна 201*, если не существует таких i, j и k, что $i < j < k$ и $a_j < a_k < a_i$. Например, последовательность $\langle 0, 1, 0, 2, 3 \rangle$ избегает паттерна 201, в то время как $\langle 0, 1, 2, 3, 1, 0, 2 \rangle$ не избегает паттерна 201, так как при $i = 4$, $j = 6$, $k = 7$ имеем $a_j = 0 < a_k = 2 < a_i = 3$.

Вам дано целое число n. Найдите количество восходящих последовательностей длины n, избегающих паттерна 201.

Формат входных данных

Единственная строка содержит целое число n ($1 \leqslant n \leqslant 15$).

Формат выходных данных

Выведите единственное число — количество восходящих последовательностей длины n, избегающих паттерна 201.

Примеры

стандартный ввод	стандартный вывод
3	5
5	52
14	65369590

Замечание

В первом примере, существует пять восходящих последовательностей длины 3, избегающих паттерна 201: $\langle 0, 0, 0 \rangle$, $\langle 0, 0, 1 \rangle$, $\langle 0, 1, 0 \rangle$, $\langle 0, 1, 1 \rangle$, $\langle 0, 1, 2 \rangle$.

Во втором примере, существует 53 восходящие последовательности длины 5, и все они избегают паттерна 201, кроме $\langle 0, 1, 2, 0, 1 \rangle$.

Задача B. Неисправный замок (версия для дивизиона 2)

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Вы — высококвалифицированный специалист в области разминирования. И вот очередной террорист поставил перед вами очередную загадку. Вы сделали практически всё возможное и невозможное, и, чтобы обезвредить бомбу, вам осталось сделать только одно действие — открыть кодовый замок. Замок состоит из двух вращающихся дисков, каждый из которых содержит все целые числа от 0 до $m-1$ включительно, упорядоченные по возрастанию. Для открытия замка нужно выставить сверху определённый код. При повороте диска вперёд число наверху диска увеличивается на 1 (за исключением случая, когда оно равно $m-1$ — тогда оно заменяется на 0). Аналогично, при повороте диска назад число наверху диска уменьшается на 1, за исключением случая, когда оно равно 0 — тогда оно переходит в $m-1$.

Вы видите текущее состояние замка и знаете обе цифры кода. К сожалению, замок неисправен — когда вы крутите какой-либо диск вперёд или назад, другой диск поворачивается в том же направлении.

Возможно, что это даже поможет вам быстрее открыть замок; возможно, что при таком дефекте замок открыть не удастся. Ваша задача — определить, можно ли открыть замок, и, если можно — за какое минимальное число поворотов он открывается.

Формат входных данных

Первая строка входа содержит одно целое число m ($2 \leqslant m \leqslant 100$), задающее диапазон чисел на дисках. Вторая строка содержит два целых числа a_1 и a_2 ($0 \leqslant a_i < m$) — числа наверху замка в момент начала работы. Третья строка содержит два целых числа b_1 и b_2 ($0 \leqslant b_i < m$) — числа, которые должны быть наверху первого и второго диска соответственно для того, чтобы открыть замок.

Формат выходных данных

Если замок открыть невозможно, выведите -1. Иначе выведите одно целое число — наименьшее количество поворотов диска, требуемое для того, чтобы открыть замок.

Примеры

стандартный ввод	стандартный вывод
6 5 2 0 3	1
7 4 1 1 5	3
10 7 7 3 5	-1

Замечание

В первом примере самый быстрый способ открыть замок — один раз повернуть диски вперёд.

Во втором примере из условия самый быстрый способ открыть замок — повернуть диски три раза назад.

В третьем примере числа на дисках остаются одинаковыми, как бы вы ни вращали диски, таким образом, замок открыть невозможно.

Задача C. Циклические сдвиги (версия для дивизиона 2)

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Пусть $s = s_1 s_2 \dots s_k$ — строка длины k. Для любого целого i от 0 до $k-1$, включительно, определим i-й *циклический сдвиг* строки s как строку $s_{i+1} s_{i+2} \dots s_k s_1 \dots s_i$. Например, 4-й циклический сдвиг строки "wellplayed" — это "playedwell", в то время как 0-й циклический сдвиг строки "metro" — это сама строка "metro".

Определим функцию $f(s)$, которая, для строки длины k, равна такому i, что i-й циклический сдвиг s лексикографически минимален среди всех её циклических сдвигов. Если существует несколько таких i, то $f(s)$ равно минимальному среди них. Например, $f(\text{“acabbac”}) = 2$, а $f(\text{“cabcab”}) = 1$.

Определим функцию $g(s)$, которая, для строки длины n, равна сумме $f(s_1 s_2 ... s_k) \cdot 10^{k-1}$ по всем k от 1 до n, включительно.

Найдите значение $g(s)$ для данной строки s.

Формат входных данных

На вводе содержится непустая строка s, состоящая из строчных латинских букв.

Длина s не превышает 9.

Формат выходных данных

Выведите единственное число — значение $g(s)$.

Примеры

стандартный ввод	стандартный вывод
aab	0
acabbac	2522200
bababa	111110
abacaba	6440200
z	0

Замечание

В первом примере, $f(\text{“a”}) = 0$, $f(\text{“aa”}) = 0$ и $f(\text{“aab”}) = 0$. Следовательно, $g(\text{“aab”}) = 0$.

Вот список значений $f(s_1 s_2 \dots s_k)$ для второго примера:

- $f(\text{“a”}) = 0$;
- $f(\text{“ac”}) = 0$;
- $f(\text{“aca”}) = 2$;
- $f(\text{“acab”}) = 2$;
- $f(\text{“acabb”}) = 2$;
- $f(\text{“acabba”}) = 5$;
- $f(\text{“acabbac”}) = 2$.

Таким образом, $g(\text{“acabbac”}) = 2 \cdot 10^2 + 2 \cdot 10^3 + 2 \cdot 10^4 + 5 \cdot 10^5 + 2 \cdot 10^6 = 2522200$.

Задача D. Распределение призовых (версия для дивизиона 2)

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Недавно прошёл наиболее популярный в Зурумбии онлайн-турнир по спортивному программированию Zurumbia Open. Призовой фонд турнира составляет x зурумбийских рублей.

Всего в турнире было 5 участников, они некоторым образом распределили места от первого до пятого, при этом гарантируется, что никакие два участника не показали одинакового результата.

К сожалению, распределение призовых между участниками на данный момент неизвестно. Известно только, что каждый участник получает целое неотрицательное количество зурумбийских рублей и ни один участник, занявший худшее место (место с более высоким номером) не получает больше, чем участник, занявший лучшее место (место с более низким номером).

Некоторые участники — ваши друзья. Вы решили вычислить наименьшую возможную суммарную долю призового фонда, которая гарантированно достанется им при любом корректном распределении призовых, а также вычислить распределение призов, при котором этот минимум достигается. Напишите программу, отвечающую на эти вопросы.

Формат входных данных

Первая строка входа содержит одно целое число x ($1 \leqslant x \leqslant 25$) — призовой фонд в зурумбийских рублях. Вторая строка содержит 5 подряд идущих латинских букв. i-я из этих букв равна "`Y`", если участник, занявший i-е место — ваш друг, и "`N`" в противном случае.

Формат выходных данных

Выведите две строки.

Первая строка содержит наименьшую возможную сумму в зурумбийских рублях, которая достанется вашим друзьям.

Вторая строка должна содержать невозрастающую последовательность из пяти целых чисел — распределение призов для мест с 1 по 5, соответственно, при котором вашим друзьям достаётся сумма, выведенная вами в первой строке. Сумма всех выведенных во второй строке чисел должна быть равна x.

Если таких последовательностей несколько, выведите любую.

Пример

стандартный ввод	стандартный вывод
23 YNYNN	10 5 5 5 4 4

Замечание

В распределении из примера к задаче ваш друг, занявший первое место, получит 7 зурумбийских рублей, занявший третье место — 3 зурумбийских рубля, итого в сумме ваши друзья получат 10 зурумбийских рублей. Можно доказать, что при любом корректном распределении призовых ваши друзья получат не менее 10 зурумбийских рублей.

Задача E. Эксклюзивная тренировка

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мебибайт

Вы — менеджер «Сквош Ит», элитного клуба по сквошу. В вашем клубе n игроков, i-й из них имеет клубный рейтинг r_i и приятность p_i.

Вы собираетесь провести эксклюзивную тренировку с одним из ваших игроков в качестве ведущего и некоторыми другими игроками с более низким рейтингом в качестве слушателей. Пронумеруем ближайшие дни таким образом, что завтра имеет номер 1, послезавтра — номер 2, и так далее. Узнав о ваших планах, каждый игрок клуба предоставил вам отрезок дней $[a_i; b_i]$ такой, что этот игрок сможет посетить тренировку только в том случае, если она будет проведена в один из дней с a_i по b_i, включительно.

Вы хотите, чтобы тренировка была не только полезной для присутствующих, но и приятной. Конечно, было бы слишком грубо не приглашать кого-либо под предлогом недостаточной его приятности. Вместо этого вы можете установить *верхнюю границу* рейтинга приглашённых, подразумевая, что тренировка предназначена для менее опытных участников.

Для каждого игрока i, вы хотите найти способ организовать тренировку с ним в качестве ведущего таким образом, чтобы суммарная приятность приглашённых игроков была как можно выше. Вы можете выбирать любой день между a_i и b_i, включительно, и любую границу рейтинга менее r_i. После этого вы пригласите на тренировку всех, кто может её посетить в данный день и имеет рейтинг не выше выбранной вами верхней границы (а также, конечно, самого игрока i). Обратите внимание, что вы можете даже выставить верхнюю границу рейтинга равной 0 — в этом случае никто, кроме ведущего, приглашён не будет, однако в некоторых случаях это может быть приятнее, чем приглашать *вон того грубияна*.

Формат входных данных

Первая строка содержит целое число n ($1 \leqslant n \leqslant 2 \cdot 10^5$) — количество игроков в «Сквош Ит». Каждая из следующих n строк содержит четыре целых числа r_i, p_i, a_i и b_i ($1 \leqslant r_i \leqslant 10^6$; $-10^6 \leqslant p_i \leqslant 10^6$; $1 \leqslant a_i \leqslant b_i \leqslant 10^6$) — клубный рейтинг, приятность и отрезок номеров дней i-го игрока, соответственно.

Клубные рейтинги всех игроков попарно различны.

Формат выходных данных

Для каждого игрока в порядке ввода выведите строку, содержащую максимальную возможную суммарную приятность приглашённых на тренировку, если этот игрок будет выбран её ведущим.

Пример

стандартный ввод	стандартный вывод
4 15 22 2 5 13 -7 5 8 9 -5 3 7 12 13 5 6	30 1 -5 13

Замечание

В примере, если тренировка будет проведена в день 5, ведущим её будет первый игрок, а приглашены будут все с рейтингом не выше 12, то приглашение на тренировку получат первый, третий и четвёртый игроки, суммарная приятность которых равна 30.

Задача F. Фруктовая игра

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Апфельманн и Бананенфрау — хорошие друзья. Сегодня Апфельманн принёс несколько яблок, а Бананенфрау — несколько бананов. Также они нашли кокос.

Друзья решили сыграть во фруктовую игру. Они расположили свои яблоки, бананы и кокос в ряд на столе. Игроки ходят по очереди, первым ходит Апфельманн.

Яблоко или банан считается *вкусным*, если между этим яблоком или бананом и кокосом не лежит ни одного другого фрукта.

На своём ходу, Апфельманн должен взять вкусное яблоко со стола и съесть его. Если на столе отсутствуют вкусные яблоки, Апфельманн пропускает свой ход. Аналогично, на своём ходу, Бананенфрау должна взять вкусный банан со стола и съесть его. Если на столе отсутствуют вкусные бананы, Бананенфрау пропускает свой ход.

Игрок, который съедает все свои фрукты раньше соперника, объявляется победителем игры.

По данному исходному расположению фруктов на столе определите, кто выиграет игру, если оба игрока сыграют оптимально и будут стараться победить.

Формат входных данных

Первая строка содержит единственное число t ($1 \leqslant t \leqslant 10^4$) — количество тестов.

Каждая из следующих t строк содержит последовательность прописных латинских букв "A", "B" и "C", задающую исходное расположение фруктов в ряду на столе в порядке слева направо. "A" обозначает яблоко, "B" обозначает банан, а "C" обозначает кокос. На столе есть как минимум одно яблоко, как минимум один банан и ровно один кокос.

Суммарная длина входных последовательностей не превышает 10^6.

Формат выходных данных

Для каждого теста выведите единственную строку, содержащую имя победителя игры — "Apfelmann" или "Bananenfrau".

Пример

стандартный ввод	стандартный вывод
3 AAACBB CABAB BBBBCBA	Bananenfrau Apfelmann Bananenfrau

Замечание

В первом примере Апфельманн и Бананенфрау по очереди едят яблоки и бананы с разных сторон от кокоса. Бананов меньше, чем яблок, поэтому Бананенфрау победит.

Во втором примере игроки едят яблоки и бананы по очереди слева направо. Апфельманн съест все яблоки раньше.

В третьем примере Апфельманну придётся пропустить первый ход. Далее у Бананенфрау будет выбор съесть банан либо слева, либо справа от кокоса. Как только Бананенфрау съест правый банан, Апфельманн съест единственное яблоко и выиграет. Для Бананенфрау будет лучше сначала съесть все бананы слева от кокоса, заставляя Апфельманна пропускать свои ходы, а далее выиграть игру, съев правый банан.

Задача G. Жадный тренер

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Некоторая тестирующая система содержит n проблемсетов, пронумерованных от 1 до n. Каждый проблемсет может быть использован для ровно одной тренировки. Иногда тренер хочет провести тренировку для команды из трёх своих студентов. Для этого тренеру необходимо назначить на тренировку такой проблемсет, что ни один из студентов в данной команде ещё не участвовал в тренировке на том же самом проблемсете. Если это невозможно, тренировка не проводится.

Рассмотрим следующие две жадные стратегии назначения проблемсетов на тренировки.

Первая стратегия, которую мы назовём *стратегией A* — назначить на тренировку проблемсет с минимальным номером среди тех, которые ещё не видел ни один из участников команды.

Вторая стратегия, которую мы назовём *стратегией B* — назначить на тренировку проблемсет, на котором ранее было проведено наибольшее число тренировок, и который в то же время ещё не видел ни один из участников команды. Если есть несколько таких проблемсетов, стратегия B выбирает среди них проблемсет с наименьшим номером.

Мы хотим выяснить, правда ли, что одна из стратегий строго лучше другой. Во-первых, либо сообщите, что это невозможно, либо найдите такое число n и такую последовательность составов команд, что стратегия A позволяет назначить проблемсеты на все тренировки, в то время как стратегия B не позволяет назначить проблемсет на хотя бы одну тренировку. Во-вторых, тот же самый вопрос задаётся относительно ситуации, в которой стратегия B позволяет назначить проблемсеты на все тренировки, а стратегия A — нет.

Формат входных данных

Единственная строка содержит целое число t ($0 \leqslant t \leqslant 2$).

Если $t = 0$, вам разрешается вывести либо -1, либо любую корректную последовательность составов команд.

Если $t = 1$, вы должны найти ситуацию, в которой стратегия A позволяет назначить проблемсеты на все тренировки, а стратегия B — нет, либо сообщить, что такая ситуация невозможна.

Если $t = 2$, вы должны найти ситуацию, в которой стратегия B позволяет назначить проблемсеты на все тренировки, а стратегия A — нет, либо сообщить, что такая ситуация невозможна.

В первом тесте $t = 0$, во втором тесте $t = 1$, в третьем тесте $t = 2$.

Формат выходных данных

Если требуемая ситуация невозможна, выведите -1.

В противном случае выведите два числа n и q ($1 \leqslant n \leqslant 100$; $1 \leqslant q \leqslant 10^4$) — количество проблемсетов и количество тренировок. Каждая из следующих q строк должна содержать три непустых строки a_i, b_i и c_i — идентификаторы студентов в команде. Эти строки должны соответствовать командам, участвующим в тренировках в хронологическом порядке. Идентификаторы студентов должны состоять из строчных латинских букв и цифр. Разные идентификаторы обозначают разных студентов. Каждая команда должна состоять из трёх различных студентов.

Гарантируется, что если требуемая ситуация возможна, то возможна и такая, в которой $n \leqslant 100$ и $q \leqslant 10^4$.

Пример

стандартный ввод	стандартный вывод
0	5 4 eatmore maxbuzz winger cerealguy eatmore niyaznigmatul cerealguy niyaznigmatul tourist qwerty787788 tourist vartem

Замечание

В примере, если тренер будет следовать любой из двух стратегий, первая и третья тренировки будут проведены на проблемсете 1, а вторая и четвёртая — на проблемсете 2.

Задача H. Очень составные перестановки

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Положительное целое число x называется *составным*, если оно имеет строго более двух положительных целых делителей. Например, числа 4, 30 и 111 — составные, а 1, 7 и 239 — нет.

Целочисленная последовательность $p = \langle p_1, p_2, \ldots, p_n \rangle$ называется *перестановкой длины* n, если она содержит каждое целое число от 1 до n, включительно, ровно по разу.

Мы будем называть перестановку $p = \langle p_1, p_2, \ldots, p_n \rangle$ *очень составной*, если для каждого i от 1 до n, включительно, сумма первых i элементов p (то есть, $p_1 + p_2 + \ldots + p_i$) является составной.

Дано целое число n. Найдите очень составную перестановку длины n.

Формат входных данных

Единственная строка содержит целое число n ($1 \leqslant n \leqslant 100$).

Формат выходных данных

Если не существует очень составных перестановок длины n, выведите единственное число -1. В противном случае выведите n целых чисел $p_1, p_2, \ldots, p_n$ таких, что $p = \langle p_1, p_2, \ldots, p_n \rangle$ — очень составная перестановка.

Если существует несколько очень составных перестановок длины n, вы можете вывести любую из них.

Примеры

стандартный ввод	стандартный вывод
13	9 13 6 5 3 2 8 4 1 12 11 10 7
2	-1

Замечание

В первом примере первый элемент перестановки, 9, — составной, сумма первых двух элементов перестановки, $9+13 = 22$, — составная, сумма первых трёх элементов перестановки, $9+13+6 = 28$, — составная, и так далее.

Во втором примере существует только две перестановки необходимой длины, $\langle 1, 2 \rangle$ и $\langle 2, 1 \rangle$, и ни одна из них не является очень составной.

Задача I. Инверсии в лексикографическом порядке (версия для дивизиона 2)

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Инверсией в перестановке $p = \langle p_1, p_2, \ldots, p_n \rangle$ называется пара целых чисел (i, j) такая, что $i < j$ и $p_i > p_j$.

Рассмотрим лексикографический порядок на целых числах. При таком упорядочивании целые числа сравниваются лексикографически как строчки цифр. Например, 628 идёт раньше 7, 239 идёт раньше 271, а 42 идёт раньше 427.

Вам дано целое число n. Отсортируем все целые числа от 1 до n, включительно, в лексикографическом порядке. Получим перестановку p длины n, в которой p_1 — лексикографически наименьшее число от 1 до n (на самом деле, $p_1 = 1$ для любого n), p_2 — второе в лексикографическом порядке, и так далее.

Сколько инверсий содержит p?

Формат входных данных

Единственная строка содержит целое число n ($1 \leqslant n \leqslant 99$).

Формат выходных данных

Выведите единственное число — количество инверсий в p.

Примеры

стандартный ввод	стандартный вывод
11	16
42	225
5	0

Замечание

Действительно, в первом примере, $p = \langle 1, 10, 11, 2, 3, 4, 5, 6, 7, 8, 9 \rangle$ содержит 16 инверсий.

Задача J. Пробежка по парку

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мебибайт

Внешний Парк очень хорош для пробежек. В парке есть n полянок, удобно пронумерованных от 1 до n. Полянки соединены m тропинками, i-я тропинка соединяет полянки a_i и b_i и имеет длину c_i метров. По всем тропинкам можно двигаться в обоих направлениях. Главный вход в парк расположен на полянке 1.

Ваши подруги решили вместе пробежаться по парку. Каждая из них подготовила свой собственный маршрут, начинающийся со входа и идущий между полянками по тропинкам.

В конце пробежки все ваши подруги хотят одновременно оказаться на полянке n с приятным кафе. Не все они правильно спланировали свой маршрут: некоторые маршруты не заканчиваются на полянке n. Кроме того, маршруты могут иметь различную длину. К счастью, все ваши подруги бегают с одинаковой скоростью.

Вы решили помочь им и написать программу, которая предоставит каждой из них *продлённый* маршрут — а именно маршрут, начинающийся с той же последовательности полянок, что и её исходный, но заканчивающийся на полянке n и имеющий такую же длину в метрах, что и все остальные продлённые маршруты. Обратите внимание, что и исходные, и продлённые маршруты могут проходить по одной и той же тропинке сколько угодно раз.

Формат входных данных

Первая строка содержит два целых числа n и m ($2 \leqslant n \leqslant 50$; $1 \leqslant m \leqslant \frac{n(n-1)}{2}$) — количество полянок и тропинок в парке, соответственно. Каждая из следующих m строк содержит три целых числа a_i, b_i и c_i ($1 \leqslant a_i, b_i \leqslant n$; $a_i \neq b_i$; $1 \leqslant c_i \leqslant 10^6$) — номера полянок, соединённых i-й тропинкой, и её длина, соответственно.

Следующая строка содержит целое число k ($2 \leqslant k \leqslant 50$) — количество ваших подруг. Каждая из следующих k строк содержит целое число l_i ($2 \leqslant l_i \leqslant 50$) — количество полянок в маршруте i-й вашей подруги, за которым следуют l_i целых чисел $g_{i,1}, g_{i,2}, \ldots, g_{i,l_i}$ ($1 \leqslant g_{i,j} \leqslant n$) — номера полянок в маршруте.

Все неупорядоченные пары (a_i, b_i) различны. Для любого i, $g_{i,1} = 1$. Для любого j от 1 до $l_i - 1$, существует тропинка между полянками $g_{i,j}$ и $g_{i,j+1}$.

Формат выходных данных

Если невозможно продлить все маршруты в соответствии с условием задачи, выведите единственное число -1.

В противном случае выведите k строк, i-я из которых содержит целое число p_i ($p_i \geqslant l_i$), за которым следуют p_i целых чисел $h_{i,1}, h_{i,2}, \ldots, h_{i,p_i}$ ($1 \leqslant h_{i,j} \leqslant n$) — номера полянок в i-м продлённом маршруте.

Для любого i, h_{i,p_i} должно быть равно n. Для любого j от 1 до $p_i - 1$, должна существовать тропинка между полянками $h_{i,j}$ и $h_{i,j+1}$. Для любого j от 1 до l_i, $h_{i,j}$ должно быть равно $g_{i,j}$. Все продлённые маршруты должны иметь одинаковую длину в метрах.

Сумма всех p_i не должна превышать $2 \cdot 10^6$. Гарантируется, что если возможное продление маршрутов существует, то существует также и продление с суммой всех p_i, не превышающей $2 \cdot 10^6$.

Пример

стандартный ввод	стандартный вывод
4 4 1 2 10 2 3 30 2 4 30 1 4 100 3 2 1 2 3 1 2 3 2 1 4	5 1 2 4 2 4 5 1 2 3 2 4 2 1 4

Задача K. Соблюдай дистанцию!

Имя входного файла:	*стандартный ввод*
Имя выходного файла:	*стандартный вывод*
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мебибайт

Вы участвуете в скандально известном ТВ-шоу. Участникам шоу часто приходится выполнять странные задания, и вы не оказались исключением...

Перед вами стоят несколько цветных кубиков. Ваша задача состоит в следующем: ведущий объявляет свой любимый цвет среди тех, которые присутствуют на кубиках. Ваша задача — переставить кубики как можно быстрее так, что их расстановка станет *красивой* относительно названного ведущим цвета.

Занумеруем кубики слева направо последовательными целыми числами, начинающимися с единицы. Пусть кубиков цвета c ровно k штук, и расположены они на позициях $p_1 < p_2 < \ldots < p_k$. Расстановка кубиков называется *красивой* относительно цвета c, если соседние кубики этого цвета находятся на одинаковых расстояниях друг от друга, то есть выполняется следующее равенство: $p_2 - p_1 = p_3 - p_2 = \ldots = p_k - p_{k-1}$. Заметим, что если $k \leqslant 2$, то расстановка всегда является красивой относительно цвета c.

При перестановке кубиков вам разрешено делать следующее действие: взять два любых (не обязательно соседних) кубика разных цветов и поменять их местами.

Вы уже видите первоначальное положение кубиков, но вы не знаете, какой цвет назовёт ведущий. Вы хотите подготовиться к выполнению задания и найти для каждого цвета наименьшее количество перестановок кубиков, требующееся для того, чтобы получить расстановку, красивую относительно этого цвета.

Формат входных данных

Первая строка входа содержит одно целое число t ($1 \leqslant t \leqslant 2\,500$) — количество тестовых примеров.

В следующих t строках заданы тестовые примеры. Каждый тестовый пример представляет собой непустую строку, состоящую из строчных латинских букв, задающих изначальную расстановку кубиков слева направо. Одинаковым буквам соответствуют одинаковые цвета, разным — разные.

Суммарная длина всех строк во входе не превосходит 250 000.

Формат выходных данных

Для каждого тестового примера выведите в отдельной строке последовательность целых чисел, разделённых пробелами. Для каждого цвета в порядке от “a” до “z”, присутствующего в тестовом примере, выведите наименьшее количество перестановок, требующееся для того, чтобы получить расстановку, красивую относительно этого цвета.

Пример

стандартный ввод	стандартный вывод
1 vvrrcvrvvr	0 1 2

Замечание

В примере из условия, расстановка уже является красивой относительно цвета “c”. Чтобы сделать расстановку красивой относительно цвета “r”, достаточно поменять местами первый и третий кубики, получив в итоге расстановку “rvvrcvrvvr”. Чтобы сделать расстановку красивой относительно цвета “v”, достаточно, например, поменять местами первый и пятый кубики, а затем второй и седьмой кубики, получив расстановку “crrrvvvvvr”.