"A" Епрестановка

Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт

Назовем *епрестановкой* такую перестановку z чисел от 1 до n, которая меняет первые два элемента местами: $z = [2, 1, 3, 4, ..., n]$.

Пусть у вас епрестановка z и перестановка $p = [p_1, p_2, ..., p_n]$ чисел от 1 до n. Рассмотрим n различных предметов, расположенных на позициях, пронумерованных от 1 до n. Предметы можно переставлять в соответствии с перестановками: после применения перестановки q, предмет, который находится на позиции j, перемещается на позицию q_j для каждого j от 1 до n.

Задано m пар целых чисел a_i и b_i. Необходимо для каждого i выяснить, возможно ли предмет, который находился на позиции a_i, переставить на позицию b_i, используя только перестановку p и епрестановку z. Для достижения цели p и z можно применять в любом порядке и сколько угодно раз.

Например, если $n = 4$, а $p = [1, 4, 3, 2]$, то элемент с позиции 4 можно переставить на позицию 1 (для этого, например, можно применить p — после этого предмет с позиции 4 окажется на позиции 2, а затем применить z), а элемент с позиции 3 на позицию 4 переместить нельзя, поскольку и p и z оставляют его на месте.

Формат входных данных	Первая строка содержит два целых числа n и m — количество элементов в перестановке и количество запросов ($2 \le n \le 10^5$, $1 \le m \le 10^5$). Следующая строка содержит n целых чисел p_i — перестановку p . Каждая следующая строка содержит по два целых числа a_i и b_i (числа a_i и b_i лежат в диапазоне от 1 до n).
Формат выходных данных	Для каждого запроса выведите в отдельной строке «`Yes`», если можно переставить предмет с позиции a_i на позицию b_i, иначе выведите «`No`».

Примеры

Входные данные	Выходные данные
4 2 1 4 3 2 4 1 3 4	Yes No

"B" Коронация

Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт

Во Флатландии приближается великий праздник! Скоро пройдет коронация ее нового правителя. Как несложно догадаться, коронация пройдет в столице. Все было бы просто, если бы во Флатландии была только одна столица. Однако, есть целых два города, одинаково претендующих на звание столицы, и до самого конца будет неизвестно, в котором из них пройдет праздник. Поэтому, необходимо дать возможность жителям каждой из столиц доехать до другой.

Транспортная система Флатландии устроена следующим образом. По дорогам страны можно доехать из каждого города в каждый, причем, единственным способом. По каждой дороге можно ездить в обе стороны. Дороги во Флатландии очень низкого качества, и для каждой дороги известно, сколько машин по ней может проехать до того, как она развалится и по ней станет невозможно ездить.

Для того, чтобы на праздник могло попасть как можно больше людей, решено было построить еще одну дорогу, соединяющую два города. Однако, так как транспортная инфраструктура столиц и так достаточно развита, ни один из городов, соединенных новой дорогой, не должен быть столицей. Но при этом она может дублировать уже существующую дорогу. Так как праздник важный, дорога будет построена качественная, и по ней сможет проехать сколь угодно много машин. Вам требуется найти максимальное количество машин, которые смогут проехать между двумя столицами после постройки новой дороги.

Формат входных данных	В первой строке заданы числа n ($4 \le n \le 10^5$), c_1 и c_2 ($1 \le c_1, c_2 \le$ n, $c_1 \ne c_2$) — количество городов во Флатландии и номера столиц. Далее, в $n - 1$ строке задано описание дорог. Каждая дорога задается целыми числами a, b и c ($1 \le a, b \le n$, $a \ne b$, $1 \le c \le 1000$) — номера городов, соединяемых этой дорогой и количество машин, которые смогут по ней проехать до того, как она развалится, соответственно.
Формат выходных данных	Выведите единственное число — максимальное количество машин, которые смогут доехать из одной столицы в другую после постройки не затрагивающей столицы дороги.

Примеры

Входные данные	Выходные данные
4 1 4 1 2 10 1 4 10 3 4 5	15

"C" Социофоб

Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт

Далеко не всем людям приятно постоянно находиться в толпе. Многие любят уединение и даже готовы за него платить. Именно поэтому ОАО «Радостные Железные Дороги» ввело на своем сайте продажи билетов новую услугу, называющуюся «Социофоб». Услуга необходима для того, чтобы дать возможность каждому пассажиру ехать в купе с наименьшим возможным количеством соседей. Суть ее заключается в следующем.

Пусть билеты продаются в некоторый вагон, в котором часть купе уже заполнена. Если в какой-то момент в этот вагон покупает билет очередной пассажир, ему продается место в самом незаполненном купе, то есть в купе, в котором количество людей не больше, чем во всех остальных. Если таких купе несколько, то выбирается купе с наименьшим номером. Если же пассажир из какого-то купе сдает свой билет, после чего разница между количеством людей в этом купе и в наиболее заполненном купе становится равной хотя бы двум, то из наиболее заполненного купе на освободившееся место пересаживается пассажир, который раньше других купил билет, то есть пассажир с наименьшим номером.

Если наиболее заполненных купе несколько, то из них выбирается купе, ближайшее к тому, в котором был сдан билет. А если и таких несколько, то выбирается купе с наименьшим номером.

По данным о том, как пассажиры покупали и сдавали билеты, требуется вывести итоговое распределение пассажиров по купе. Гарантируется, что каждый раз когда какой-либо пассажир покупает билет, в вагоне есть хотя бы одно свободное место.

Формат входных данных	В первой строке заданы три целых числа m, n и k ($1 \le m \le 200000$, $1 \le n, k \le 50000$) — количество операций покупки или сдачи билетов, количество купе в вагоне и количество мест в каждом купе, соответственно. В следующих m строках задана последовательность покупок и сдач билетов. Если строка содержит единственный символ «+», то это означает, что в вагон купил билет очередной пассажир, причём номер пассажира равен порядковому номеру плюса во входном файле. Если строка имеет вид «− id», где id — целое число от одного до m, то это означает, что пассажир с номером id сдал свой билет. Гарантируется, что на этот момент у этого пассажира есть билет.
Формат выходных данных	Выведите n строк, где первое число i-й строки l_i означает, сколько билетов продано в i-ом купе, далее следуют l_i чисел — номера пассажиров, которые будут ехать в i-ом купе, выведенные в порядке возрастания. Гарантируется, что ответ всегда существует.

Примеры

Входные данные	Выходные данные
4 2 2 + + + - 2	1 3 1 1
8 3 2 + + + + + - 2 - 5	1 4 1 1 2 3 6

"D" Монеты

Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт

Анк-Морпорк — огромный город с оживленной торговлей. В его порт заглядывают корабли из самых дальних стран. В связи с этим в обращении циркулирует большое количество монет разных государств. Для упрощения торговых отношений, у каждой монеты есть фиксированный обменный курс к официальной валюте города — Анк-Морпоркскому доллару. Будем называть этот курс номиналом монеты.

Подобное разнообразие приводит к тому, что даже самые опытные горожане иногда

путают монеты между собой. Вот и волшебник Ринсвинд недавно заплатил за новую шляпу T долларов вместо положенных S. К тому моменту, когда он это обнаружил, лавочника уже и след простыл.

Ринсвинд считает, что он какую-то монету принял за другую, скажем, дублон за стерлинг. Каждый раз, когда он думал, что отсчитывает стерлинг, на самом деле он отдавал дублон. При этом настоящих стерлингов он не платил. Например, Ринсвинд мог думать, что заплатил дублон и два стерлинга, а на самом деле он отдал три дублона.

Ринсвинду интересно, сколько существует пар монет таких, что приняв первую монету за вторую, он мог заплатить ровно T долларов вместо S, как он посчитал.

Например, если в обращении есть монеты номиналом один, два и три доллара, и Ринсвинд заплатил десять долларов вместо девяти, то он мог это сделать, если принял монеты номиналом два за монеты номиналом один и заплатил, например, четыре монеты номиналом два и одну монету номиналом два, которую принял за монету номиналом один. Также, он мог принять монеты номиналом три за монеты номиналом два и заплатить семь монет номиналом один и одну номиналом три, которую принял за монету номиналом два.

А если в обращении есть монеты номиналом два, три и четыре доллара, и Ринсвинд заплатил одиннадцать долларов вместо десяти, то он мог это сделать, если принял монеты номиналом три за монеты номиналом два и заплатил две монеты номиналом четыре и одну монету номиналом три, которую принял за монету номиналом два.

Формат входных данных	Первая строка содержит три целых числа S, T и n ($1 \le S, T \le 10000$; $S \ne T$; $1 \le n \le 200$) — стоимость шляпы, сумма, потраченная Ринсвиндом, и число монет, соответственно. Следующая строка содержит n целых чисел a_1, a_2, ..., a_n — номиналы монет ($1 \le a_i \le 10000$). Никакие две монеты не имеют одинаковый номинал.
Формат выходных данных	Выведите число пар монет таких, что приняв первую монету за вторую, Ринсвинд мог заплатить T долларов вместо S.

	Входные данные	**Выходные данные**
Примеры	9 10 3 1 2 3	2
	10 11 3 2 3 4	1

"E" Разбор строки

Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт

При разработке алгоритмов и систем, занимающихся поиском в интернете, важную роль играет часть программы, отвечающая за разбор выражений. За время существования поисковых систем было придумано большое количество различных алгоритмов, решающих задачи этого класса. Они отличаются друг от друга временем работы,

сложностью написания и количеством потребляемых ресурсов.

Рассмотрим задачу проверки того, состоит ли некоторая строка из слов, принадлежащих к некоторому словарю. Одним из самых простых алгоритмов, решающих эту задачу, является жадный алгоритм. Каждый раз из строки удаляется максимальный ее префикс, присутствующий в словаре как отдельное слово. На достаточно большом множестве строк, являющихся конкатенацией слов из словаря, этот алгоритм успешно завершает свою работу и разбивает строку на слова. Кроме того, этот алгоритм крайне прост в реализации. Однако, существуют и строки, на которых он выдаст неправильный результат. Например, если в качестве словаря взять список всех слов, существующих в английском языке, то строка «workingrass» не будет распознана корректно, так как сначала из нее будет удален префикс «working», а после этого остаток «rass» разбить на нужные слова уже невозможно. В то же время, эту строку можно представить как конкатенацию слов «work», «in» и «grass».

При разработке различных программ далеко не всегда приходится иметь дело со словарем английского языка. Иногда словари достаточно специфичны, и описанный алгоритм будет корректно разбивать всевозможные строки, являющиеся конкатенациями слов из словаря. Вам необходимо проверить, является ли набор слов (словарь) таким, и, если не является, привести пример строки, для которой разбиение на слова из словаря существует, но описанным алгоритмом оно не будет найдено. Если таких примеров несколько, необходимо найти кратчайший, то есть тот, длина которого не больше, чем у остальных. Если и таких несколько, выведите любой.

Формат входных данных	Первая строка содердит одно целое число n ($1 \le n \le 250$) — количество слов в словаре. В следующих n строках перечислены слова, лежащие в словаре. Каждое слово непусто и состоит только из строчных букв латинского алфавита. Длина каждого слова не превышает 500 символов.
Формат выходных данных	Если строки, отвечающие описанным в условии требованиям, существуют, выведите любую из них, длина которой не превышает длины остальных. Выведенная вами строка должна состоять только из строчных символов латинского алфавита. В противном случае выведите «Good vocabulary!».

Примеры

Входные данные	Выходные данные
3 ab cd abcd	Good vocabulary!
3 aba ab ac	abac

"F" Принц

Ограничение по времени	2 секунды
Ограничение по памяти	256 мегабайт

После долгих приключений Принц невероятно близок к спасению Принцессы. Ему осталось лишь преодолеть коридор, наполненный ловушками.

Коридор представляет собой бесконечную в обе стороны прямую. В начальный момент времени Принц находится в точке 0. В x метрах от него находится дверь, ведущая в комнату Принцессы. За одну секунду Принц может пройти один метр в любом направлении или остаться на месте.

Использовав Пески времени, Принц выяснил, что в коридоре расположены n ловушек. Ловушки работают следующим образом: i-ая ловушка появится через a_i секунд и исчезнет через t_i секунд после появления. Ловушка занимает часть коридора с l_i по r_i метров (отсчет ведется от Принца в направлении Принцессы), и если Принц окажется строго в занимаемой ловушкой области, его ожидает неминуемая гибель. Принц может безопасно находиться на краю ловушки, а также сразу проходит в дверь, даже если в этот же момент там появляется ловушка.

Ваша задача состоит в том, чтобы по описанию ловушек узнать, через какое минимальное время Принц сможет добраться до двери, ведущей в комнату Принцессы.

Формат входных данных	Первая строка содержит два целых числа n и x ($0 \le n \le 1000$; $1 \le x \le 10^5$) — число ловушек и позиция двери. Далее следуют n строк, содержащих описание ловушек. Описание состоит из четырех целых чисел a_i, t_i, l_i и r_i ($1 \le a_i, t_i \le 10^6$; $-10^6 \le l_i < r_i \le 10^6$) — время появления и продолжительность жизни ловушки, а также положение ее левого и правого краев. Ловушки могут пересекаться.
Формат выходных данных	Если Принц не может добраться до двери, выведите «`Impossible`». Иначе выведите одно целое число — минимальное время, через которое Принц сможет добраться до заветной двери.

Примеры

Входные данные	Выходные данные

`3 2`
`1 1 -1 2`
`3 2 1 3`
`6 1 0 2`
	`6`

`1 2` `1 1 -2 2`	`Impossible`