Factory. Фабрика

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

На фабрику привезли новое оборудование. В связи с этим, была произведена аттестация рабочих, и выяснилось, что не все рабочие достаточно квалифицированы для обслуживания новых станков. При этом на каждом станке могут работать не более двух рабочих.

Руководство завода хочет узнать, какое максимальное количество рабочих можно поставить за новые станки.

Формат входного файла

Первая строка входного файла содержит три целых числа n, m и k ($0 \le k \le 10\,000$), где n и m — количество рабочих и станков соответственно ($0 \le n, m \le 100$).

Последующие k строк содержат по два числа каждая — номер рабочего и номер станка, на котором он может работать.

Формат выходного файла

Первая строка выходного файла должна содержать одно натуральное число — максимальное количество рабочих, которых можно поставить за новые станки.

Пример

factory.in	factory.out
5 2 6 2 1 5 1 1 2 3 1 4 1 1 1	3

Floyd. Pink Floyd

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

Группа *Pink Floyd* собирается дать новый концертный тур по всему миру. По предыдущему опыту группа знает, что солист *Роджер Уотерс* постоянно нервничает при перелетах. На некоторых маршрутах он теряет вес от волнения, а на других — много ест и набирает вес.

Известно, что чем больше весит Роджер, тем лучше выступает группа, поэтому требуется спланировать перелеты так, чтобы вес Роджера на каждом концерте был максимально возможным.

Группа должна посещать города в том же порядке, в котором она дает концерты. При этом между концертами группа может посещать промежуточные города.

Формат входного файла

Первая строка входного файла содержит три натуральных числа n, m и k — количество городов в мире, количество рейсов и количество концертов, которые должна дать группа соответственно ($n \le 100$, $m \le 10\,000$, $2 \le k \le 10\,000$). Города пронумерованы числами от 1 до n.

Следующие m строк содержат описание рейсов, по одному на строке. Рейс номер i описывается тремя числами b_i, e_i и w_i — номер начального и конечного города рейса и предполагаемое изменение веса Роджера в миллиграммах ($1 \le b_i, e_i \le n$, $-100\,000 \le w_i \le 100\,000$).

Последняя строка содержит числа $a_1, a_2, \ldots, a_k$ — номера городов, в которых проводятся концерты ($a_i \ne a_{i+1}$). В начале концертного тура группа находится в городе a_1.

Гарантируется, что группа может дать все концерты.

Формат выходного файла

Первая строка выходного файла должна содержать число l — количество рейсов, которые должна сделать группа. Вторая строка должна содержать l чисел — номера используемых рейсов.

Если существует такая последовательность маршрутов между концертами, что Роджер будет набирать вес неограниченно, то первая строка выходного файла должна содержать строку “`infinitely kind`”.

Пример

floyd.in	floyd.out
4 8 5 1 2 -2 2 3 3 3 4 -5 4 1 3 1 3 2 3 1 -2 3 2 -3 2 4 -10 1 3 1 2 4	6 5 6 5 7 2 3
4 8 5 1 2 -2 2 3 3 3 4 -5 4 1 3 1 3 2 3 1 -2 3 2 -3 2 4 10 1 3 1 2 4	infinitely kind

Negcycle. Отрицательный цикл

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

Дан взвешенный ориентированный граф. Требуется определить, содержит ли он цикл отрицательного веса.

Формат входного файла

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно ($n \le 1\,111$, $m \le 11\,111$).

Следующие m строк содержат описание ребер по одному на строке. Ребро номер i описывается тремя числами b_i, e_i и w_i — номерами концов ребра и его весом соответственно ($1 \le b_i, e_i \le n$; $-100\,000 \le wi \le 100\,000$).

В графе могут быть кратные ребра и петли.

Формат выходного файла

Первая строка выходного файла должна содержать `yes`, если граф содержит цикл отрицательного веса и `no` — в противном случае.

Пример

negcycle.in	negcycle.out
4 4 2 1 -4 1 2 1 3 4 2 2 3 3	yes
4 6 2 1 4 1 2 1 3 4 2 2 3 3 1 1 2 1 2 2	no

King. Король (*)

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

В Тридесятом царстве, Тридевятом государстве жил-был король. И было у короля n сыновей. В Тридесятом царстве жили n прекрасных девушек, и король знал, какие девушки нравятся каждому сыну (поскольку сыновья были молодыми и безшабашными, то им могли нравиться несколько девушек одновременно).

Однажды король приказал своему советнику подобрать для каждого сына прекрасную девушку, на которой тот сможет жениться. Советник выполнил приказ и подобрал для каждого сына для женитьбы прекрасную девушку, которая ему нравилась. Разуемеется, каждая девушка может выйти замуж только за одного из сыновей.

Посмотрев на список невест, король сказал: «Мне нравится этот список, но я хочу знать для каждого сына список всех девушек, на которых он может жениться. Разумеется, при этом все сыновья также должны иметь возможность жениться на девушках, которые им нравятся».

Эта задача оказалася для советника слишком сложной. Помогите ему избежать казни, решив ее.

Формат входного файла

Первая строка входного файла содержит число n — количество сыновей ($1 \le n \le 2\,000$). Следующие n строк содержат списки прекрасных девушек, которые нравятся сыновьям. В начале идет k_i — количество девушек, которые нравятся i-му сыну. Затем идут k_i чисел — номера девушек. Сумма k_i не превышает $200\,000$.

Последняя строка входного файла содержит список, составленный советником — n различных чисел от 1 до n: для каждого сына — номер прекрасной девушки, на которой он может жениться. Гарантируется что список корректен — то есть каждому сыну нравится выбранная для него девушка.

Формат выходного файла

Выходной файл должен содержать n строк. Для каждого сына выведите l_i — количество различных девушек, на которых он может жениться. После этого выведите l_i чисел — номера девушек в произвольном порядке.

Пример

king.in	king.out
4 2 1 2 2 1 2 2 2 3 2 3 4 1 2 3 4	2 1 2 2 1 2 1 3 1 4