Non-Cooperative Multi-Commodity Network Flows

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 mebibytes

In the late 1980s, ****** started as a small company providing communication services and equipment. With time, ****** had grown to a worldwide IT giant with a large scope of end products. Still, communication remains one of the core subjects of interest for the company. Today, ****** faces global scope challenges and seeks breakthrough solutions that can be used by everyone.

You are probably familiar with the *network flow* problem: find the maximum flow through a network. Despite its natural interpretation for vehicle transport networks or communication networks, you will rarely find direct usage of it in these settings. There are two main issues basic network flow settings are not addressing:

- **Commodities**. In practice, we usually have different types of flow needed to coexist in a single network; these types are called *commodities.* The main thing is that while the flow of a single type cancels itself when going through the edge in the opposite direction, for flows of different types, there are no such mechanisms. In transport networks, commodities are different drivers that want to get to different destinations or different goods that need to be transported through the network. In communication networks, commodities are packets of different connections.
- **Non-cooperative goals**: in case we are optimizing the logistics of a single company, we can set up a single goal for different commodities. In case we are trying to optimize traffic light schedules in a city, we are dealing with thousands to millions of drivers trying to reach their own destinations.

Modern approaches to model the latter setup are based on the assumption that *every driver acts in a way that minimizes his travel time* and result in a sort of *Nash equilibrium* that can be stated as follows: *the system converges to a state when no driver can achieve better travel time by altering their own path while the rest of the drivers do not change their path.* In the context of transport systems, this state is usually called *Wardrop equilibrium*, and it is known to exist in a continuous setting with some reasonable travel time / load dependency. In this challenge, we are asking you to find an equilibrium state in a similar communication network scenario.

The network consists of n nodes connected by m communication channels. The i-th communication channel provides the possibility of transferring data in one direction from node u_i to node v_i and has the following characteristics.

- Integer c_i: the soft threshold for the channel capacity measured in Kbit/s (kilobits per second). Note that c_i is not a strict capacity like in classic network flow problems: it is allowed to go beyond the capacity and make life worse for everyone (which unfortunately happens a lot in real life applications).
- Integers ω_i and α_i: *normal* cost and *overload* cost, respectively. If the total flow f over the channel is at most c_i, the cost to use this channel is ω_i per 1 Kbit/s. Otherwise, the cost is $\omega_i + \alpha_i \cdot (f - c_i)$ per 1 Kbit/s.

We have to transfer q commodities over the network. For the i-th commodity, we have to transfer f_i Kbit/s from source node s_i to destination node t_i.

Given the descriptions of communication channels and commodities, find a way to fully transfer all commodities from their sources to their respective destinations. You can split each commodity into up to 100 integer parts, and transfer each part via its own path. The goal is to get as close to an equilibrium as you can.

Input

The first line contains three integers: the number of nodes n, the number of channels m, and the number of commodities q ($2 \le n \le 10^4$; $1 \le m \le 3 \cdot 10^4$; $1 \le q \le 100$).

The next m lines describe one-way communication channels. The i-th of these lines contains five integers: the starting node of the channel u_i, the end node v_i, the soft threshold for channel capacity c_i, the normal cost ω_i, and the overload cost α_i ($1 \le u_i, v_i \le n$; $0 \le c_i \le 4000$; $0 \le \omega_i, \alpha_i \le 10^4$).

The next q lines describe commodities. The i-th of these lines contains three integers: the source node s_i, the destination node t_i, and the data rate f_i ($1 \le s_i, t_i \le n$; $s_i \ne t_i$; $1 \le f_i \le 10^4$).

The sum of data rates over all commodities is bounded as $\sum_{i=1}^{q} f_i \le 5 \cdot 10^5$.

For each commodity, there exists at least one path from source to target.

Output

Print the way to transfer each of the q commodities, in order. For commodity i:

On the first line, print the number of paths p_i you want to use ($1 \le p_i \le 100$). On each of the next p_i lines, print a single path. The description of path j starts with two integers: the number of channels ℓ_{ij} on the path and the flow f_{ij} through the path ($1 \le \ell_{ij} \le m$; $1 \le f_{ij} \le f_i$). Then follow ℓ_{ij} integers $e_{ij1}, \ldots, e_{ij\ell_{ij}}$: the numbers of communication channels that form the path, in order ($1 \le e_{ijk} \le m$). Each path should go from s_i to t_i. The sum of flows over all paths should be equal to the data rate of the commodity: $\sum_{j=1}^{\ell_{ij}} f_{ij} = f_i$.

Scoring

There are 2 example tests and 77 main tests. To pass a test, your solution has to print any answer that satisfies all the conditions above. A solution that passes both examples is checked on all main tests, and gets a score from 0 to 100 independently for each test. The total score for your solution is the sum of the scores for each main test, rounded to the nearest integer.

The teams are ranked according to their score: the more, the better. Your score is the highest total score of all the solutions you send during the contest. Ties are resolved by the time of submission: the earlier, the better. To prevent testing queue, there is a submission limit: during the contest, you can send **at most 50 solutions**.

The score for each test is calculated as follows:

Let x_{ik} be the flow of the i-th commodity through channel k. The cost of passing 1 Kbit/s through channel k is then

$$\mathcal{J}_k(x) = \omega_k + \alpha_k \cdot \max\left(0, \sum_{i=1}^{q} x_{ik} - c_k\right),$$

and the total cost of commodity i is

$$\Phi_i(x) = \sum_{k=1}^{m} \mathcal{J}_k(x) \cdot x_{ik}.$$

Now, let $\Delta_i(x)$ be the set of all flows obtained by rerouting 1 Kbit/s. The penalty for the i-th commodity is then

$$\sigma_i(x) = \max\left\{0, \Phi_i(x) - \min_{y \in \Delta_i(x)} \Phi_i(y)\right\},$$

and the total penalty is the average penalty over all commodities:

$$\sigma(x) = \frac{1}{q} \sum_{i=1}^{q} \sigma_i(x).$$

Your goal is to make the total penalty $\sigma(x)$ as small as you can. It is known from theory that the global minimum of the function σ is 0. If your answer is flow x, the score is as follows (rounded to 0.5):

$$\mathit{Score} = \max\left(0, 100 - 6\log(1 + \sigma(x))\right).$$

Examples

standard input	*standard output*
2 2 1 1 2 10 0 10 1 2 10 20 5 1 2 100	2 1 36 1 1 64 2
6 10 5 1 2 3 1 15 1 2 5 7 10 2 5 0 3 4 5 6 3 5 1 2 6 2 10 0 2 3 2 5 5 3 5 6 5 0 3 6 9 10 1 1 4 5 40 1 4 3 7 30 1 3 6 10 1 3 10 1 4 6 2 6 7 4 6 12	2 1 7 8 2 3 7 4 3 2 3 1 6 2 5 2 6 2 2 9 10 1 1 6 9 1 1 7 5 2 2 6 10 8 3 6 10 7 4

Note

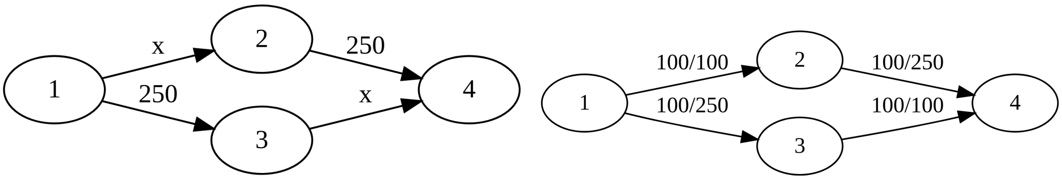

Figure 1: On the left, the initial network configuration: four connections, two of them having a constant cost of 250 ($c = 0$, $\omega = 250$, $\alpha = 0$) and two of them having a linear cost of x ($c = 0$, $\omega = 0$, $\alpha = 1$). On the right, the optimal distribution of flow for both single commodity and several commodity cases.

Here is a complex example of what might happen in a network: the infamous Braess' paradox. The configuration contains four nodes with four connections like in Figure 1. Two connections have constant cost and two connections have linear cost. Suppose that we have either a single commodity with demand 200 or 200 commodities with demand 1 to transfer from node 1 to node 4. In both cases, the optimal distribution is 100/100 between the two paths, resulting in a cost of 350 per unit of flow.

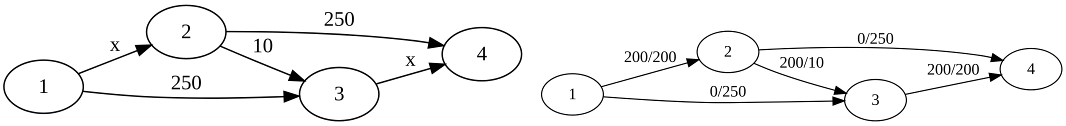

Figure 2: The same configuration but with an additional connection $2 \to 3$ added. In case of multiple commodities, the optimal distribution results in a worse situation than the distribution without this connection.

Now, look at Figure 2. Here, we added an intermediate connection with a low cost. For the case of multiple commodities, it is beneficial to change the path with cost $250 + x$ into the path with cost $10 + 2x$. As a result, all the flow is now accumulated over the $1 \to 2 \to 3 \to 4$ path, resulting in the cost of 410 per unit of flow: worse than without the intermediate connection. The situation is different for the case of a single commodity, it is not beneficial to use the intermediate connection: if we change it for a single unit of flow, the cost of that unit will decrease, but the cost of all other units of flow will increase, resulting in a worse situation, like we discussed above.

We expect an ideal solution to fall for Braess' paradox in case of multiple commodities, and to resolve it in case of a single commodity.