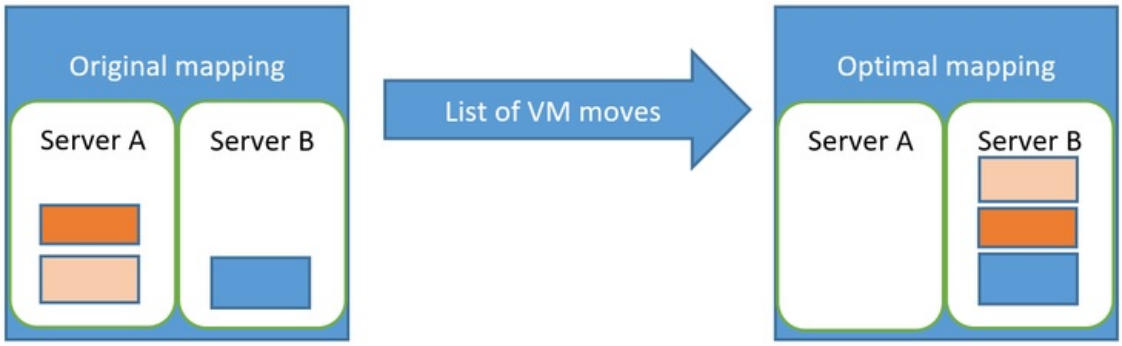
NERC Challenge 2020: Marathon

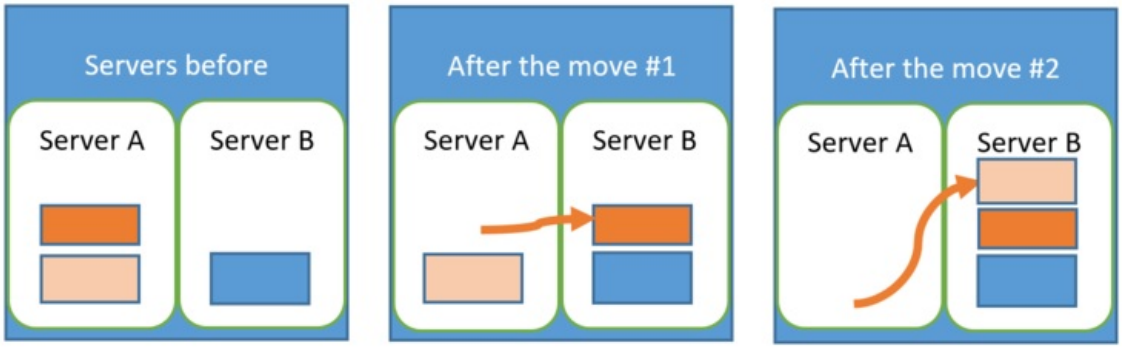
A. Fixing the Cloud

time limit per test: 6 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

We are operating a fleet of N servers and M virtual machines, and some smart artificial intelligence has determined that the current mapping of VMs to servers is not optimal. The best mapping of virtual machines (VMs) was also found by the AI, but it fails to produce a good enough schedule of VM moves to achieve this ideal mapping. Can you help?



To change the mapping, you can move VM to any other server if there are enough resources. After one or several moves of the VM, it must be placed as planned:



Each server is characterized by two values: the i-th server has sc_i CPU cores and sm_i GB RAM. Each VM is characterized by two values: the j-th VM consumes vc_j cores and vm_j GB RAM.

You can move multiple VMs in the cloud at the same time. Parallel moves take the same time and are grouped into "steps". Each move is characterized by three values: source server, target server, and VM to move.

Constraints:

1. You do not control the order of moves inside a step. Before each step, the system checks if there are enough resources for each server for every VM it currently has and all arriving in the upcoming step. The release of previous resources (removed VMs) will happen between the steps.
2. The network is a bottleneck, so no more than 2 parallel moves per server is possible (for example, if a server receives 3 new VMs and 1 old VM moves somewhere, this will take at least 2 steps). In particular, if there are N servers in the cloud, you may have $N \times 2 \div 2 = N$ moves in a step, at most.

It is supposed that CPU cores and memory consumed by VM are allocated in the target server before the step started and deallocated in the source server only after the step is finished. It means that VMs consumes resources while transferring.

The goal is to find the schedule with minimal time to complete and avoid redundant migrations, if possible. The total score to be minimized is defined as follows:

$$Score = 1000 \cdot \log_{10}\left(Total\_steps \times Memory\_moved\_in\_GB + 1\right)$$

- $2 \le N \le 1000$, $1 \le M \le 100\,000$;
- Each server i has sc_i CPU cores and sm_i GB RAM, $100 \le sc_i \le 500$, $200 \le sm_i \le 1000$;
- Each VM j consumes vc_j cores and vm_j GB RAM, $1 \le vc_j \le 200$, $1 \le vm_j \le 500$;
- Each server may host any number of VMs if its resource capacity is not violated.

The total score will be the sum of each test score. Each test score is rounded to 3 digits after the decimal points according to

mathematical rules. Each test run without a solution (or failure during execution) will be penalized, $penalty\_score = 10\,000$ will be used for this case. The jury guarantees that the solution exists for each test.

The problem has two suites of tests: preliminary and main. The preliminary test suite consists of 30 tests. This test suite is used to test your solutions during the contest (that is, immediately after submitting a solution). When the contest is over, then for each participant **the last submission** will be selected (which passed at least one test). This particular submission will be tested on the main test suite (other submissions will be ignored). The main suite consists of a preliminary suite plus another 20 tests (so it contains a total of $30 + 20 = 50$ tests). After testing on the main test suite, the final standings table will be built, on the basis of which the final results of the contest will be announced.

Input

The first line of each test contains two integers — N and M ($2 \le N \le 1000$, $1 \le M \le 100\,000$).

Next, N lines contain server descriptions. The i-th of them contains two integers — sc_i and sm_i ($100 \le sc_i \le 500$, $200 \le sm_i \le 1000$).

Next, M lines contain VM descriptions. The j-th of them contains two integers — vc_j and vm_j ($1 \le vc_j \le 200$, $1 \le vm_j \le 500$).

Next, M lines follow, each containing two integers old_j and new_j ($0 \le old_j, new_j < N$) — server index where j-th VM located in the original mapping and server index where it should be located in the optimal mapping.

Output

In the first line print a single integer S ($0 \le S \le 3 \cdot 10^6$) — total steps count.

Then print S line groups with step descriptions. Each description should start with a line containing a single integer k_i ($1 \le k_i \le M$) — the number of parallel VM moves in the current step. Next print k_i lines with each VM move description — triplets of zero-indexed old server index, new server index, and VM index.

The jury guarantees that the solution exists for each test.

Examples

input
2 1 100 500 200 1000 50 200 0 1
output
1 1 0 1 0

input
3 3 100 350 200 480 150 720 80 300 20 50 150 480 0 1 2 0 1 2
output
3 1 2 0 1 1 1 2 2 1 0 1 0

input
5 6 500 1000 100 200 500 1000 500 500 500 1000 200 500 50 50 200 500 50 150 50 50 50 50 0 2 2 0 3 4 1 2 2 0 0 1

output

```
3
2
2 0 1
2 0 4
3
0 1 5
0 2 0
3 4 2
1
1 2 3
```