ETH Zurich
Institut für Theoretische Informatik
Prof. Dr. Angelika Steger
Prof. Dr. Emo Welzl
Prof. Dr. Peter Widmayer

Algorithms Lab

Exercise – *Kingdom Defence*

In a land far far away lies the mysterious kingdom of swamps. For centuries it has been a peaceful country little heeded by the lands around it. But times change and the ever-growing greed of its neighbors has brought the kingdom of swamps to the brink of war. While the peaceful inhabitants of the kingdom deeply despise war, some cities have been wise enough to start training soldiers early and now have well equipped garrisons. Other cities only command negligible defences.

The king of the swamps is a wise and gentle ruler. He wants to avoid open war at all costs. With its abundance of impenetrable swamps, his kingdom is well protected and the king hopes that if the most important strategic locations are strengthened and enough military presence is openly displayed, no neighbor will dare to attack. Moving battalions of soldiers around through the swamps is difficult however. There are only very narrow paths which can only be used in one direction as it is not always possible to make room for oncoming traffic. The king has asked you for help.

Problem You are given a map of the (directed) paths between the most important strategic locations in the kingdom of swamps. Every location has a number of soldiers available and every location needs a certain number of soldiers to defend itself. In order to display military presence, each path has to be travelled by some minimum number of soldiers. Also, because soldiers on the move constantly need to supply themselves with food, every path has a maximum number of soldiers that can walk along it in total.

You have to decide whether it is possible to satisfy all these requirements by moving battalions of soldiers around in the kingdom.

Input The first line of the input contains the number $t \leq 30$ of test cases. Each of the t test cases is described as follows:

- It starts with a single line containing two integers `l q`, separated by a space, denoting
  - l, the number of locations ($1 \leq l \leq 500$).
  - p, the number of paths ($1 \leq p \leq l^2$).
- The next l lines each give the details of one location.
  The i-th such line contains two space-separated integers $1 \leq g_i, d_i \leq 10^6$, the number of soldiers stationed at location i and the number of soldiers that location i needs to defend itself, respectively. Note that $g_i > 0$ for all locations.
- After that, there are p lines, each specifying one path in the kingdom.
  Each path is given by four integers $0 \leq f_j, t_j < l$ and $0 \leq c_j \leq C_j \leq 10^6$. The j-th path is one-way and goes from location f_j to location t_j (zero-based indexing, f_j and t_j can be equal!). The path needs to be traversed by at least c_j and by at most C_j soldiers. Note that a single soldier may use the same path multiple times, but each time counts as one traversal.

Output For every test case output a single line containing the word `yes`, if the soldiers can be moved such that during the move enough military presence is displayed along every path and after moving every location is well defended, and the word `no` otherwise.

Points There are four groups of test sets, worth 100 points in total.

1. For the first group of test sets, worth 20 points, you may assume that there are no minimum requirements for the paths, i.e. $\forall j = 1, \ldots, p\colon\ c_j = 0$.
2. For the second group of test sets, worth 20 points, you may assume that each location has only incoming paths or only outgoing paths, i.e. there are no f_i, t_k with $f_i = t_k$.
3. For the third group of test sets, worth 30 points, there are no additional assumptions.
4. For the fourth group of test sets, worth 30 points, there are no additional assumptions.

Corresponding sample test sets are contained in `test`i`.in/out`, for $i \in \{1, 2, 3\}$.

Sample Input

```
4
3 3
5 2
2 3
1 3
0 1 0 3
1 2 0 3
2 0 0 2
4 4
6 1
5 1
1 6
2 6
0 2 2 5
0 3 3 6
1 2 2 4
1 3 2 4
3 3
5 2
2 3
1 3
0 1 2 3
1 2 2 3
2 0 1 2
3 4
4 2
1 3
1 1
0 1 2 6
1 2 0 4
2 1 3 6
1 0 4 5
```

Sample Output

```
yes
no
no
yes
```