Problem I: Impossible Install

Time limit: 4 seconds

Irina was hired as an unpaid intern to handle IT at the Olympic Games. Fortunately, there are a bunch of Python projects that already handle everything. They just have to be installed.

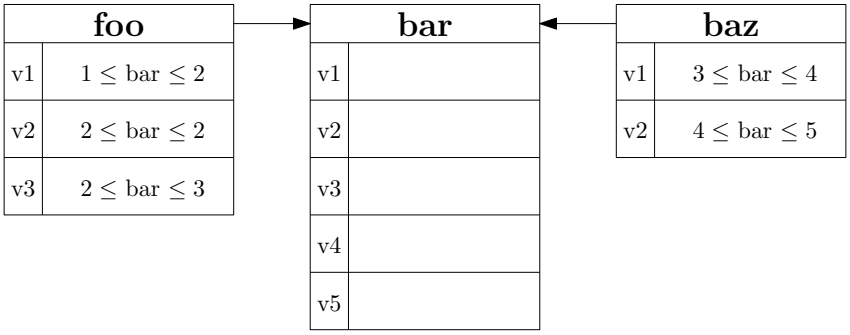

Figure I.1: Illustration of the Sample Input 1. The projects are called foo, bar, and baz here.

Garry, the intern who was tasked with the setup before, quit unexpectedly but left some notes. "What is going on?", "This is insane!" and "QUIT WHILE YOU CAN!!!". However, some older notes seem to be more helpful and give an idea about the setup. There are software projects and dependencies between them that form a directed acyclic graph without multi-edges. Software projects have multiple versions. A version of a project specifies a lower and an upper bound on the version of each dependency. The bounds on the dependencies are weakly increasing over the versions of a project.

Help Irina pick a version for each project such that all dependencies are satisfied.

Input

The input consists of:

- One line with an integer n ($1 \leq n \leq 10^5$), the number of projects. The projects are numbered from 1 to n.
- n sections, the pth of which describes project p:
  - One line with two integers v_p and d_p ($1 \leq v_p \leq 10^9$, $0 \leq d_p < n$), the number of versions and the number of dependencies of project p.
  - d_p dependencies of project p, each consisting of three lines:
    - One line with an integer q ($1 \leq q \leq n$, $q \neq p$), indicating that project p depends on project q.
    - One line with v_p integers $\ell_1, \ldots, \ell_{v_p}$ ($1 \leq \ell_i \leq v_q$, $\ell_i \leq \ell_{i+1}$ for each i), where ℓ_i is the minimum version of project q that is required by version i of project p.
    - One line with v_p integers $r_1, \ldots, r_{v_p}$ ($1 \leq r_i \leq v_q$, $r_i \leq r_{i+1}$, $\ell_i \leq r_i$ for each i), where r_i is the maximum version of project q that is supported by version i of project p.

It is guaranteed that $\sum_{p=1}^{n} v_p \cdot d_p \leq 10^6$ and that the dependency graph has no cycles. The dependencies of each project are distinct projects.

Output

If there is no way to install everything, then output a line with "`impossible`". Otherwise, output "`possible`" followed by n integers giving a version of each project such that all dependencies are satisfied.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1

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

Sample Output 1

```
possible
3 3 1
```

Sample Input 2

```
3
1 1
3
1
1
1 1
3
3
3
3 0
```

Sample Output 2

```
impossible
```

Sample Input 3

```
2
1 1
2
400000000
600000000
1000000000 0
```

Sample Output 3

```
possible
1 400000000
```