# Party Planning

Lea has a lot of things to do. Her birthday is coming up and she wants to host a big party. There are cakes to be baked, decoration to be set up, invitations to be sent, drinks to be cooled, and so on. As Lea likes to plan ahead and dislikes stress very much, she wants to find a perfect schedule to finish all her prep work as quickly and as calmly as possible. Luckily for her, she has lots of close friends (as many as she needs) that propose to help her set up the perfect birthday party. After all, they are the guests and want to enjoy the party as much as Lea does.

Unfortunately, not all needed tasks can be done simultaneously: For example, Lea cannot bake a cake unless all ingredients have been bought before, she cannot decorate the room unless it has been cleaned, and she cannot buy the right amount of drinks until she has checked all the answers on her RSVP cards. But from many years of experience she knows some things. She knows the exact amount of time needed to finish any certain task, and she knows the dependencies of all the needed tasks, i.e. which task has to be done before another can be started. Furthermore, she knows that the first task has to be to write a checklist, and the last task to be completed is to cross off all items on it. But as Lea's friends are not nearly as organized as she is, she needs to present them with an exact schedule so that everyone knows what he or she has to do. Can you help Lea to prepare her party as quickly as possible?

## Input

The first line of the input contains an integer $t$. $t$ test cases follow, each of them separated by a blank line.

Each test case starts with a single line containing an integer $n$, which denotes the number of tasks (w.l.o.g. numbered from 1 to $n$). $n$ lines follow. The $i$-th line consists of a sequence of integers $p_i$, $s_i$, and $j_{i,1}, \ldots, j_{i,s_i}$, separated by spaces. $p_i$ denotes the number of time units needed to finish task $i$, $s_i$ is the number of tasks that are direct successors of task $i$, i.e. tasks that need task $i$ to be finished before they can be started. The sequence $j_{i,1}, \ldots, j_{i,s_i}$ lists all direct successors of task $i$, in no specific order.

The first task to be done (the source) is task 1, it is a predecessor of all other tasks; the last task (the sink) is task $n$, it is a successor of all other tasks.

## Output

For each test case, output one line containing "Case #$i$: $x$" where $i$ is its number, starting at 1, and $x$ is the total number of time units needed for the prep work.

## Constraints

- $1 \leq t \leq 20$
- $1 \leq n \leq 1000$
- $1 \leq p_i \leq 1000$ for all $1 \leq i \leq n$
- $0 \leq s_i \leq n - 1$ for all $1 \leq i \leq n$
- $i < j_{i,k} \leq n$ for all $1 \leq i \leq n$ and $1 \leq k \leq s_i$

**Sample Input 1**

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

5
6 4 2 3 4 5
7 3 3 4 5
3 2 4 5
2 1 5
2 0

2
3 1 2
1 0
```

**Sample Output 1**

```
Case #1: 11
Case #2: 20
Case #3: 4
```

**Sample Input 2**

```
4
5
6 4 2 3 4 5
4 3 4 3 5
7 1 5
8 1 5
7 0

8
1 7 2 3 4 5 6 7 8
7 6 3 4 7 5 6 8
5 2 6 8
5 4 6 5 7 8
1 2 6 8
2 2 7 8
1 1 8
3 0

6
4 5 2 3 4 5 6
6 4 5 3 4 6
4 2 5 6
8 2 5 6
2 1 6
5 0

8
4 7 2 3 4 5 6 7 8
6 2 6 8
3 4 7 4 6 8
3 3 5 7 8
4 1 8
7 2 7 8
7 1 8
8 0
```

**Sample Output 2**

```
Case #1: 25
Case #2: 20
Case #3: 25
Case #4: 32
```