Problem E. Stack Management

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 10 points	Solve E-small
Large input 30 points	Solve E-large

Problem

You are playing a solitaire game in which there are **N** stacks of face-up cards, each of which initially has **C** cards. Each card has a *value* and a *suit*, and no two cards in the game have the same value/suit combination.

In one move, you can do one of the following things:

1. If there are two or more cards with the same suit that are on top of different stacks, you may remove the one of those cards with the smallest value from the game. (Whenever you remove the last card from a stack, the stack is still there — it just becomes empty.)
2. If there is an empty stack, you may take a card from the top of any one of the non-empty stacks and place it on top of (i.e., as the only card in) that empty stack.

You win the game if you can make a sequence of moves such that eventually, each stack contains at most one card. Given a starting arrangement, determine whether it is possible to win the game.

Input

The first line of the input gives the number **P** of premade stacks that will be used in the test cases. Then, **P** lines follow. The i-th of those lines begins with an integer $\mathbf{C_i}$, the number of cards in the i-th of those premade stacks, and continues with $\mathbf{C_i}$ ordered pairs of integers. The j-th of these ordered pairs has two integers $\mathbf{V_{ij}}$ and $\mathbf{S_{ij}}$, representing the value and suit of the j-th card from the top in the i-th premade stack.

Then, there is another line with one integer **T**, the number of test cases. **T** test cases follow. Each case begins with one line with two integers **N** and **C**: the number of stacks, and the number of cards in each of those stacks. Then, there is one line with **N** integers $\mathbf{P_i}$, representing the indexes (starting from 0) of the test case's set of premade stacks.

Output

For each test case, output one line containing `Case #x: y`, where x is the test case number (starting from 1) and y is `POSSIBLE` if it is possible to win the game, or `IMPOSSIBLE` otherwise.

Limits

$1 \le \mathbf{T} \le 100$.
$2 \le \mathbf{P} \le 60000$.
$0 \le \mathbf{P_i} < \mathbf{P}$, for all i.
The $\mathbf{P_i}$-th premade stack has exactly **C** cards.
No two cards in a test case have the same value/suit combination.

Small dataset

$2 \le \mathbf{N} \le 4$.
$2 \le \mathbf{C_i} \le 13$, for all i.
$2 \le \mathbf{C} \le 13$.
$1 \le \mathbf{V_{ij}} \le 13$, for all i and j.
$1 \le \mathbf{S_{ij}} \le 4$, for all i and j.

Large dataset

$2 \le \mathbf{N} \le 50000$.
$2 \le \mathbf{C_i} \le 50000$, for all i.
$2 \le \mathbf{C} \le 50000$.
$4 \le \mathbf{N} \times \mathbf{C} \le 10^5$.
$1 \le \mathbf{V_{ij}} \le 50000$, for all i and j.
$1 \le \mathbf{S_{ij}} \le 50000$, for all i and j.

Sample

Input Output

```
5                Case #1: POSSIBLE
2 7 2 7 1        Case #2: IMPOSSIBLE
2 6 4 7 4
2 3 2 6 2
2 4 2 10 2
2 5 4 7 3
2
2 2
0 2
3 2
4 1 3
```

In sample case #1, there are two stacks, each of which has two cards. The first stack has a 7 of suit 2 on top and a 7 of suit 1 below that. The second stack has a 3 of suit 2 on top and a 6 of suit 2 below that.

It is possible to win the game as follows:

- Remove the 3 of suit 2 from the second stack.
- Remove the 6 of suit 2 from the second stack. This makes the second stack empty.
- Move the 7 of suit 2 to the second stack. Then the win condition is satisfied: all stacks have at most one card.

In sample case #2, there are three stacks, each of which has two cards. It is not possible to win the game in this case; the only possible move is to remove the 5 of suit 4 on top of the third stack, and this does not open up any new moves.