Problem F. Teleporters

Source file name:	F.c, F.cpp, F.java, F.py
Input:	`Standard`
Output:	`Standard`

Mankind has evolved into a pretty awesome state. We have flying cars. We now live in planets other than Earth. But many people's favorite invention is the teleporter. Teleporters are way too expensive to be owned by every single planet. However, there are some rich planets which do have them.

The problem with the current teleporting technology is that they only work in pairs. In other words, one teleporter can only teleport people to ONE specified exit. They are bidirectional, so an "entry" teleporter can also work as an "exit" teleporter. Traveling between teleporters takes no time.

Your job as a travel planner is to come up with the best route between planets. The best route is the one that takes the less amount of time. Obviously, you can use teleporters to do this.

Input

First you will be given an integer N, the number of test cases. N maps, for which you have to calculate best routes, will follow.

For each of the N maps, the first line contains two integers, $1 \leq V \leq 50$, and $T \geq 0$, where V is the number of planets in the system, and T is the number of pairs of teleporters.

After these 2 numbers, V lines will follow. The first integer in these lines represents a planet, and the next pairs of numbers represent planets which can be reached from the current one, as well as the time it takes to reach them. Note that even if there is a path from planet A to planet B, that does not necessarily mean there is a path from B to A. The numbering of the planets starts with 0.

After that, there is a line with $T * 2$ integers, which represent the teleporters. These should be read pairwise, that is, if the line looks like this:

0 1 2 3

that means there is a teleporter pair between planets 0 and 1, and another pair between planets 2 and 3. If T is zero, then this line is omitted.

Finally, there is a line with two integers, the first represents the starting planet, and the second the goal.

Output

If there is a path from the starting planet to the goal planet, print one line with the format "Case #X: time" (X starts in 1), followed by a line containing all the planets visited, in order. Each of these planets will be separated by a single whitespace.

In case there exists no path, print "Case #X: Path does not exist", and omit the next line.

Note that the minimum path will always be unique.

Also, teleporter pairs can only be used once. So, if you already used a teleporter going from planet A to B, then you cannot use that same teleporter again, not even if going from B to A.

Example

Input

```
3
4 2
0 1 10
1 2 10
2 3 10
3 0 10
3 2 2 0
1 0
3 0
0 1 10 2 20
1
2 1 2
0 2
3 0
0 1 10 2 20
1
2 1 2
1 2
```

Output

```
Case #1: 10
1 2 0
Case #2: 20
0 2
Case #3: Path does not exist
```