Problem C. **Pony Express**

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 16 points	Solve C-small
Large input 24 points	Solve C-large

Problem

It's the year 1860, and the Pony Express is the fastest mail delivery system joining the East and West coasts of the United States. This system serves **N** different cities. In each city, there is one horse (as in the expression "one-horse town"); each horse travels at a certain constant speed and has a maximum total distance it can travel before it becomes too tired to continue.

The Pony Express rider starts off on the starting city's horse. Every time the rider reaches a city, they may continue to use their current horse or switch to that city's horse; switching is instantaneous. Horses never get a chance to rest, so whenever part of a horse's maximum total distance is "used up", it is used up forever! When the rider reaches the destination city, the mail is delivered.

The routes between cities were established via complicated negotiations between company owners, lawmakers, union delegates, and cousin Pete. That means that the distances between cities do not necessarily follow common sense: for instance, they do not necessarily comply with the triangle inequality, and the distance from city A to city B might be different from the distance from city B to city A!

You are a time traveling entrepreneur, and you have brought a fast computer from the future. A single computer is not enough for you to set up an e-mail service and make the Pony Express obsolete, but you can use it to make optimal routing plans for the Pony Express. Given all data about routes between cities and the horses in each city, and a list of pairs of starting and ending cities, can you quickly calculate the minimum time necessary for each delivery? (You should treat all of these deliveries as independent; using cities/horses on one route does not make them unavailable on other routes.)

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case is described as follows:

- One line with two integers: **N**, the number of cities with horses, and **Q**, the number of pairs of stops we are interested in. Cities are numbered from 1 to **N**.
- **N** lines, each containing two integers $\mathbf{E}_i$, the maximum total distance, in kilometers, the horse in the i-th city can go and $\mathbf{S}_i$, the constant speed, in kilometers per hour, at which the horse travels.
- **N** lines, each containing **N** integers. The j-th integer on the i-th of these lines, $\mathbf{D}_{ij}$, is -1 if there is no direct route from the i-th to the j-th city, and the length of that route in kilometers otherwise.
- **Q** lines containing two integers $\mathbf{U}_k$ and $\mathbf{V}_k$, the starting and destination point, respectively, of the k-th pair of cities we want to investigate.

Output

For each test case, output one line containing `Case #x: y1 y2 ... yQ`, where x is the test case number (starting from 1) and y_k is the minimum time, in hours, to deliver a letter from city $\mathbf{U}_k$ to city $\mathbf{V}_k$.

Each y_k will be considered correct if it is within an absolute or relative error of 10^{-6} of the correct answer. See the FAQ for an explanation of what that means, and what formats of real numbers we accept.

Limits

$1 \le \mathbf{T} \le 100$.
$2 \le \mathbf{N} \le 100$.
$1 \le \mathbf{E}_i \le 10^9$, for all i.
$1 \le \mathbf{S}_i \le 1000$, for all i.
$-1 \le \mathbf{D}_{ij} \le 10^9$, for all i, j.
$\mathbf{D}_{ii} = -1$, for all i. (There are no direct routes from a city to itself.)
$\mathbf{D}_{ij} \ne 0$, for all i, j.
$\mathbf{U}_k \ne \mathbf{V}_k$, for all k.
It is guaranteed that the delivery from $\mathbf{U}_k$ to $\mathbf{V}_k$ can be accomplished with the given horses, for all k.
$\mathbf{U}_l \ne \mathbf{U}_m$ and/or $\mathbf{V}_l \ne \mathbf{V}_m$, for all different l, m. (No ordered pair of cities to

investigate is repeated within a test case.)

Small dataset

$\mathbf{D}_{ij}$ = -1, for all i, j where i + 1 ≠ j. (The cities are in a single line; each route goes from one city to the next city in line.)
$\mathbf{Q}$ = 1.
$\mathbf{U}_1$ = 1.
$\mathbf{V}_1$ = $\mathbf{N}$. (The only delivery to calculate is between the first and last cities in the line).

Large dataset

1 ≤ $\mathbf{Q}$ ≤ 100.
1 ≤ $\mathbf{U}_k$ ≤ $\mathbf{N}$, for all k.
1 ≤ $\mathbf{V}_k$ ≤ $\mathbf{N}$, for all k.

Sample

Input

```
3
3 1
2 3
2 4
4 4
-1 1 -1
-1 -1 1
-1 -1 -1
1 3
4 1
13 10
1 1000
10 8
5 5
-1 1 -1 -1
-1 -1 1 -1
-1 -1 -1 10
-1 -1 -1 -1
1 4
4 3
30 60
10 1000
12 5
20 1
-1 10 -1 31
10 -1 10 -1
-1 -1 -1 10
15 6 -1 -1
2 4
3 1
3 2
```

Output

```
Case #1: 0.583333333
Case #2: 1.2
Case #3: 0.51 8.01 8.0
```

Note that the last sample case would not appear in the Small dataset.

In Case #1 there are two options: use the horse in city 1 for the entire trip, or change horses in city 2. Both horses have enough endurance, so both options are viable. Since the horse in city 2 is faster, it is better to change, for a total time of 1/3 + 1/4.

In Case #2 there are two intermediate cities in which you can change horses. If you change horses in city 2, however, your new horse, while blazingly fast, will not have enough endurance, so you will be forced to change again in city 3. If you keep your horse, you will have the option to change horses (or not) in city 3. So, the three options, with their total times, are:

1. Change horses in both city 2 and 3 (1/10 + 1/1000 + 10/8 = 1.351).
2. Change horses just in city 3 (2/10 + 10/8 = 1.45).
3. Never change horses (12/10 = 1.2).

In Case #3, there are lots of alternatives for each delivery. The optimal one for the first delivery (city 2 to city 4) is to go to city 1 in time 10/1000, change horses, and then go to cities 2, 3 and 4, in that order, using the horse from city 1, which takes time (10 + 10 + 10) / 60.

For the second delivery (city 3 to city 1) you have no choice but to first go to city 4 which takes time 10/5. Your relatively fast horse does not have enough endurance to get anywhere else, so you need to grab the horse in city 4. You could use it to get directly to city 1 in time 15, but that would be slower than riding it to city 2 in time 6 and then using the blazingly fast horse in city 2 to get to city 1 in just 10/1000 extra time.

In the third delivery (city 3 to city 2) of Case #3 it is optimal to use the first two steps of the previous one, for a total time of 10/5 + 6 = 8.