Internet is Faulty

David has a problem. He wants to transfer a big file through the internet from his home computer to his computer at work. The size of his file is S kilobytes.

The internet consists of N computers, numbered from 1 to N. David's home computer has the number 1 and his computer at work has the number 2. Some pairs of computers are connected by different types of links (such as network cables or wi-fi). Some of these links (e.g., a satellite dish) may be unidirectional, thus for simplicity we will assume that all links are unidirectional.

The data are sent across the network in packets, each packet contains exactly one kilobyte of data. For each link David knows how faulty it is, i.e. the probability that a network packet gets from one computer to the other through the link. We assume that all transfers are independent from each other. That is, regardless of whether the previous packet was transferred successfully or not, the probability that the next one will pass through remains the same.

Since the network is faulty and the work computer might be many links away from the home computer, the transfer of David's file along even the best route between the two computers might take too long. Luckily, David has an account on some of the machines in the network. He may use these machines as temporary storage, and thus shorten the time of the transfer.

The file transfer will consist of several steps. In each step, David selects a series of links starting with a computer that already has the file and ending with a computer David has an account on. Prior to the transfer, the file is split into S packets. Then the packets are sent one after another along the chosen route. The probability that a packet successfully arrives at the destination computer is the product of probabilities that it passes all the links. If the packet is lost it is resent immediately. Each attempt to send a packet, successful or unsuccessful, takes exactly one millisecond, regardless of the number of links on the route. After the entire file is transferred, David may start another transfer from the new machine, and so on.

Problem specification

You are given the number of computers N and the file size S. For each pair of computers u, v we know the probability p(u, v) that a network packet passes successfully through the direct link from computer u to v. (The value zero means that there is no direct link from u to v.) Finally, you are given a list of servers where David has an account.

Find a way how to send David's file so that the expected transfer time is minimized, and output this expected time in milliseconds. You may assume that the expected transfer time is less than 1 000 000 000 (1 billion) milliseconds.

Input specification

The first line of the input file contains an integer T specifying the number of test cases. Each test case is preceded by a blank line.

Each test case looks as follows: The first line contains a positive integer 2<= N <=200 giving the number of computers on the internet.

N lines follow. The i-th of these lines contains N integers, each of them between 0 and 100, inclusive. These integers give the probabilities p(i, 1), p(i, 2), ..., p(i, N) in percents.

The next line contains a non-negative integer M – the number of computers on which David has an account. The following line contains M integers – the numbers of these computers. This list will always contain the integers 1 and 2 (corresponding to David's home and work computer).

The last line contains an integer S – the size of David's file in kilobytes.

Output specification

For each test case output a single line with a real number – the expected time of the transfer in milliseconds when using the best possible strategy.

Each number in the output file should have sufficiently many decimal places. We recommend printing all results rounded to seven decimal places. Your output will be considered correct if each number has an absolute or relative error less than 10^{-6}.

Example

Input:

```
2

4
0 0 40 66
0 0 0 30
40 47 0 66
0 30 66 0
4
1 2 3 4
47

5
0 1 20 0 0
0 0 0 0 0
0 0 0 50 90
0 20 0 0 0
0 0 0 90 0
3
1 2 5
10
```

Output:

```
207.897153
111.111111
```

Hint

In the second case, we have four possible strategies.

The first one is to use the direct link from 1 to 2. This link is really really faulty, and the expected time for this solution is 1000 milliseconds. The second strategy is to transfer the file from computer 1 to computer 2 using the route 1-3-4-1. The probability that a packet passes this route is 20% * 50% * 20% = 2%, and thus the expected transfer time is 500 milliseconds. The third

strategy is to use the route 1-3-5-4-2. Here the probability of successfully transfering a packet is 3.24% and the expected transfer time is roughly 308.6 milliseconds.

The optimal solution is to use the computer 5 as temporary storage. We will first transfer the file along the route 1-3-5, and then along the route 5-4-2. For each of the transfers the probability is 20% * 90% = 18%, and thus the expected time of each of the transfers is 55.5555 milliseconds.