Fitting a function to a given finite set of points sampled from an unknown function $F : \mathbb{R} \to \mathbb{R}$ is a basic problem in mathematics. Typical one is to find a linear function $\overline{F}$ that fits the sampled point best. One way to measure how well $\overline{F}$ fits the sample points is defined as follows. Suppose that F is sampled at $x_1, \ldots, x_n$, with $x_1 < \ldots < x_n$. Then the error of $\overline{F}$ is

$$error(F, \overline{F}) = \max_{1 \le i \le n} \{|F(x_i) - \overline{F}(x_i)|\}$$

Unfortunately, the function values at the sample points are not known exactly. Instead, we have a discrete probability distribution for each $F(x_i)$, that is, we have a discrete set $y_{i,1}, \ldots, y_{i,m_i}$ of possible values with associated probabilities $p_{i,j}$ such that $\mathbf{Pr}[F(x_i) = y_{i,j}] = p_{i,j}$. We define the error measure in the following natural way using the concept of the expected value:

$$error(F, \overline{F}) = \max_{1 \le i \le n} \{\mathbf{E}[|F(x_i) - \overline{F}(x_i)|]\}$$

The goal is now to find a linear function $\overline{F} = ax + b$ that minimizes the error. You must write a program that gets the sample points and their probabilities and computes the minimum error defined above.

Input

There are multiple test cases in the input. The first line of each test case contains n, the number of the sampled points ($1 \le n \le 10^5$). Next, the information of sampled points (in the increasing order) comes in n lines; one line for each sampled point. For the i-th sampled point, the line starts with two non-negative integer numbers x_i and m_i where x_i is the value at which we sample the function and m_i is the size of distribution ($1 \le m_i \le 10$ and $0 \le x_i \le 10^9$). Then it is followed by a list of m_i non-negative numbers being less than 10^9 which are the function values at x_i, and finally a list of m_i probabilities. For simplicity, each given probability p in the input is a non-negative integer less than or equal to 100. You can get the real probability by dividing p to 100. You can assume the summation of all probability numbers is equal to 100. The input terminates with a line containing '0' which should not be processed.

Output

For each test case, output a line containing the minimum error rounded to exactly one digit after the decimal point.

Sample Input

```
2
0 2 0 1 50 50
1 2 0 1 50 50
0
```

Sample Output

```
0.5
```