Market Optimization Problem

Limits: 4 sec., 1024 MiB

Imagine you own c boxes of chocolate and you want to sell all of them during the next T days. You are not interested in gaining the maximal possible profit. You are only interested in influencing the world chocolate prices as little as possible.

There are n major chocolate buyers available to you. Buyer number i is willing to buy no more than L_i chocolate boxes. On a day number t they will buy chocolate for a price of $P_{i,t}$ dollars. You know all of the values L_i for all buyers. As to the prices, you only know the values $P_{i,1}$, i.e. the prices for all buyers for the first day. The rest of the prices are calculated using the following equations:

$$P_{i,t+1} = P_{i,t} + \Delta P_{i,t}$$

$$\Delta P_{i,t} = P_{i,t}\left(1 - \frac{1}{e^{k_{i,t}\frac{Q_{i,t}}{L_{i,t}+1}}}\right) + \epsilon_{i,t} \cdot P_{i,t} + \sum_{j<i}\left(\frac{\alpha_{i,j,t}\Delta P_{j,t}}{ln(max(e, |P_{i,t} - P_{j,t}|))}\right) + \sum_{s=1}^{t-1}\beta_s Q_{i,t-s}$$

- $P_{i,t}$ — the price for the buyer number i on the day number t,
- $\Delta P_{i,t}$ — the price change for the buyer number i on the day number t,
- $Q_{i,t}$ — the number for chocolate boxes that you sell to the buyer number i on the day number t,
- $L_{i,t} = L_i - \sum_{j=1}^{t-1} Q_{i,j}$ — the maximal number of chocolate boxes you can sell to the buyer number i on the day number t,
- $k_{i,t}$ — the coefficient of market influence for the buyer number i on the day number t,
- $\epsilon_{i,t}$ — the random noise for the buyer number i on the day number t,
- $\alpha_{i,j,t}$ — the coefficient of the influence of the price change of the buyer number j to the buyer number i on the day number t,
- β_s — the coefficient of the influence of the orders placed s days ago.

Your task is to come up with the values $Q_{i,t}$ such that $0 \le Q_{i,t} \le L_{i,t}$ and $\sum_{i=1}^{n}\sum_{t=1}^{T} Q_{i,t} = c$ which describe how many boxes of chocolate you want to sell to each buyer on each day. Your goal is to have the prices on the final day to be as close to the original prices as possible. Please refer to the Scoring section for more details.

Input

The first line of the input contains three space separeted integers n, T and c: the total number of buyers, the total number of days and the total number of chocolate boxes that you have.

Each of the next n lines contain a couple of space separated integers $P_{i,1}$ and L_i.

Each of the next n lines contain T space separated floating point numbers $k_{i,t}$.

Each of the next n lines contain T space separated floating point numbers $\epsilon_{i,t}$.

Each of the next $n-1$ blocks contain values $\alpha_{i,j,t}$. The block number i (for each i from 2 to n) consists of $i-1$ lines. Each of those lines contain T space separated floating point numbers $\alpha_{i,j,t}$.

The last line contains T space separated floating point numbers β_s.

Each floating point number is given with exactly 5 digits after the decimal point.

Output

Print T lines. Line number t should contain exactly n space separated integers $Q_{i,t}$ — the number of boxes of chocolate sold on day number t to each buyer.

Constraints

$1 \le n \le 100$,
$2 \le T \le 100$,
$1 \le c \le 10^6$,
$1 \le L_i \le 10^5$,
$1 \le P_{i,1} \le 10^9$,
$|k_{i,t}| \le 10$,
$|\epsilon_{i,t}| \le 0.05$,
$|\alpha_{i,j,t}| \le 0.025$,
$|\beta_i| \le 0.06$.

Samples

Input (*stdin*)	Output (*stdout*)
2 2 7 47 70 77 110 4.47474 -7.77477 7.74747 -4.44744 0.00047 0.00074 -0.00074 -0.00047 0.00004 -0.00007 -0.00600 0.00600	2 2 1 2

Notes

Example

After the first day the price will be changed in the following way:

$$\Delta P_{1,1} = 5.58822$$

$$\Delta P_{2,1} = 9.97532$$

$$P_{1,2} = P_{1,1} + \Delta P_{1,1} = 47 + 5.58822 = 52.58822$$

$$P_{2,2} = P_{2,1} + \Delta P_{2,1} = 77 + 9.97532 = 86.97532$$

After the second day the price will be changed in the following way:

$$\Delta P_{1,2} = -6.24535$$

$$\Delta P_{2,2} = -7.44796$$

$$P_{1,3} = P_{1,2} + \Delta P_{1,2} = 52.58822 + -6.24535 = 46.34287$$

$$P_{2,3} = P_{2,2} + \Delta P_{2,2} = 86.97532 + -7.4479 = 79.52736$$

The total score:

$$\left\lfloor \frac{10 \cdot (47 + 77) - (|47 - 46.34287| + |77 - 79.52736|)}{10 \cdot (47 + 77)} \cdot 10^7 \right\rfloor = 9974318$$

Scoring

Your answer for a test case is considered invalid, if your output doesn't match the format specified in the Output section, or if one of the values $Q_{i,t}$ violates the constraints, or if the sum of all $Q_{i,t}$ is not equal to c. In this case the score for the test case is 0. If your answer is valid, then the score for the test case is calculated as following:

$$\max\left(0, \left\lfloor \frac{10 \cdot \sum_{i=1}^{n} P_{i,1} - \sum_{i=1}^{n} |P_{i,T+1} - P_{i,1}|}{10 \cdot \sum_{i=1}^{n} P_{i,1}} \cdot 10^7 \right\rfloor\right)$$

Submissions

- The execution time limit is 4 seconds per test case, and the memory limit is 1024 mebibytes.
- The code size limit is 64 kibibytes.
- The compilation time limit is 1 minute.
- There are 50 provisional test cases. Your submissions will be evaluated on the provisional set during the submission phase.
- You can submit your code once every 10 minutes, and you will get feedback with your score for each of the provisional tests.
- There will be 500 test cases in the final testing after the submission phase is over. Please note that provisional tests are **not included** in the final testing. The final results will be announced in one week.

Quick start

Check the sample solution, which tries to sell all of the chocolate boxes at the latest possible moment. The source code is available for some of the contest programming languages:

- C++
- Python
- Java
- C#

Tests

All test cases, including provisional and final sets, are generated by the generator. Please note that the generator uses the Algotester Generator library. You can check the source code for the library here: algotester_generator.h

You can use the generator for local testing by taking a few simple steps:

1. Check the generator source code
2. Save it to a file named `generator.cpp` in the folder with `algotester_generator.h`
3. Compile the source code:

   ```
   g++ generator.cpp -O2 -o generator
   ```

4. Generate a test case:

   ```
   ./generator <seed>
   ```

   This will print a test case for a specific seed to *standard output*.