

Problem D. A fish lunch

Input file: `stdin`
Output file: `stdout`
Time limit: 1 sec
Memory limit: 256 Mb

Alex decided to go on a trip. Now he is flying in the airplane, the lunchtime has just started. Lunch can be one of two kinds — meat or fish, and flight attendants ask each of the passengers, what kind of dinner he or she prefers.

But Alex knows that neither lunch of one type will not be enough for all passengers. If one of the types ends, flight attendants stop asking passengers about their preferences and just give them what remains. Alex wants to get a fish lunch. He knows that there are N passengers in the airplane, N_1 meat lunches and N_2 fish lunches. Moreover, he looked in what order does flight attendants distribute lunches, and noticed that he is I . He also has a statistics, from which we learn that fish lunches are preferred by $P\%$ of passengers. Now he wants to know, what's his chance to get a fish dinner?

Limit

$$0 \leq N_1 \leq 1000$$

$$0 \leq N_2 \leq 1000$$

$$1 \leq N \leq N_1 + N_2$$

$$1 \leq I \leq N$$

$$0 \leq P \leq 100$$

Input

The single line contains 5 integers: N , N_1 , N_2 , I and P .

Output

You should print only one number — Alex's chance to get fish lunch. Output response should be with an absolute percentage error of no more than 10^{-7} .

Example

<code>stdin</code>	<code>stdout</code>
10 5 5 10 50	0.5
10 8 7 8 60	0.9720064

Problem E. The secret code

Input file: `stdin`
Output file: `stdout`
Time limit: 1 sec
Memory limit: 256 Mb

After returning from the trip, Alex was unpleasantly surprised: his porch door had a new combination lock. Alex can not get into his house! Code lock contains N disks, each of which can be in one of M positions. There is only one correct position. Alex thoroughly inspected discs and by fingerprints and scratches determined the probability of each position for each disk. Now Alex has K attempts to guess the correct code: if he fails, his vigilant neighbors will call the police, and Alex will have a hard time persuading cops that he is not a thief, but tried to get home. Help Alex to calculate the maximum probability of getting home, not to the police.

Limit

$$1 \leq N \leq 100$$

$$1 \leq M \leq 20$$

$$1 \leq K \leq 100$$

$$0 \leq P_{ij} \leq 100$$

Input

The first line of input file contains three integers: N , M and K .

Next N lines contain M integers each: j -th number of the i -th line (P_{ij}) — the probability of a situation where i -th disc's correct position is j . Given that $\sum_{j=1}^M P_{ij} = 100$.

Output

Print a single number — Alex's chances to guess the correct code in time. Output result should have an absolute percentage error not greater than 10^{-7} .

Example

stdin	stdout
2 2 1 50 50 10 90	0.45
3 5 4 10 15 20 25 30 1 2 3 4 90 100 0 0 0 0	0.81

Problem F. Dura Lex

Input file: `stdin`
Output file: `stdout`
Time limit: 1 sec
Memory limit: 256 Mb

Alex loves to order gadgets from abroad. Unfortunately for Alex, new customs regulations has been recently introduced, and according to them he needs N passes to receive each gadget. Every pass is hard to get — to get i -th pass, Alex needs to spend D_i days in a queue, and he can only stand in a queue for requesting not more than one pass at any moment. The most annoying part is i -th pass' request can be denied with probability $P_i\%$, absolutely randomly, and moreover, if one pass request is denied, all the others get annulled automatically and Alex has to start it all over again. There is only one good thing about it — Alex can get passes in whatever order he wants.

Alex wants get a new gadget by all means. He will try to get all the passes, until he has them all. Help him select the order of requesting the passes so that average time needed to collect them all will be minimum.

Limit

$$1 \leq N \leq 10^5$$

$$0 \leq D_i \leq 1000$$

$$0 \leq P_i < 100$$

Input

The first line of input file contains an integer N .

Next N lines contain two integers each: D_i and P_i .

Output

Print N integers from 1 to N — desired order. If there are several optimal orders, print the lexicographically minimal.

Example

stdin	stdout
3 1 10 10 10 5 90	3 1 2
3 2 20 3 30 4 40	1 2 3

Problem G. A path to knowledge

Input file: `stdin`
Output file: `stdout`
Time limit: 1 sec
Memory limit: 256 Mb

Alex studies at university and goes there every day by foot. City where Alex lives, is a nonoriented graph. Alex decided to use a scientific approach when choosing the road, so he studied the map of the city and found all the shortest routes from home to university. Now every time Alex goes to the university or back, he chooses one of the routes randomly, each choice has with equal probability.

In a few days Alex noticed that he passes some crossroads more often than others. He decided to calculate how many times a day he walks by each crossroads on average. But since he is busy studying, he charged you to do that for him.

Limit

$$1 \leq N \leq 10^5$$

$$0 \leq M \leq 10^5$$

$$1 \leq A_i, B_i \leq N$$

$$1 \leq L_i \leq 10000$$

Input

The first line of input file contains two integers: N and M — the number of crossroads and roads in Alex's city.

Each of the following M lines corresponds to one street, and contains three integers A_i , B_i and L_i — number of crossroads the street connects and its length in kilometers.

Alex's house is right next to the 1st crossroads, his university — is beside the N -th. It is guaranteed that it is possible to reach university from Alex's house by roads.

Output

Print out N numbers — the average number of passes through the crossroads from first to N -th per day. Print the result with an absolute percentage error not grater than 10^{-7} . Keep in mind that Alex goes through the city twice a day — when he goes to the university and back home.

Example

stdin	stdout
3 3 1 2 1 2 3 1 1 3 1	2.0 0.0 2.0
4 4 1 2 1 1 3 1 2 4 1 3 4 1	2.0 1.0 1.0 2.0

Problem H. A granite of science

Input file: `stdin`
Output file: `stdout`
Time limit: `1 sec`
Memory limit: `256 Mb`

At the university, where Alex studies, the new semester has begun. The new semester brings to Alex N new subjects, and each subject will be held every day. Before the beginning of the semester Alex learned how many lessons scheduled for each of the objects: for i -th subject planned N_i lessons for all i one to N . In addition, some of senior students said Alex the difficulty of each of the subjects: the number of H_{ij} for all i and j denotes the difficulty of j -th lesson i -th subject.

Many teachers have not yet returned from holidays, so the lessons of some subjects will not start immediately. More specifically, the first day of lesson for each subject can be determined with equal probability of the first M days of semester. May even happen that the first day in university can be without lessons at all. Once first lessons of any subject has begun, they are held regularly, one lesson per day, until all N_i lessons will not end.

During his studies, Alex gets tired. He thought that total fatigue of one day is the square of sum of fatigue of all lessons this day. To better understand what expect, Alex wants to know total fatigue of the entire semester. Since Alex does not know exactly when lessons would start, he is interested in the average value.

Limit

$1 \leq N \leq 500$
 $1 \leq M \leq 500$
 $1 \leq N_i \leq 500$
 $0 \leq H_{ij} \leq 1000$

Input

The first line contains two integers: N and M .

Each of the next N rows corresponds to one subject and contains an integer N_i — number of lessons of the subject, — and N_i integers H_{ij} — the difficulty of each these lessons.

Output

You should print a single number — the total value of Alex's fatigue per semester. Print the answer with absolute or relative error of less than 10^{-7} .

Example

stdin	stdout
2 1 1 1 1 2	9.0
3 3 1 3 2 1 1 3 0 0 0	14.3333333

Problem I. Probable diagnosis

Input file: `stdin`
Output file: `stdout`
Time limit: `1 s`
Memory limit: `256 Mb`

As a course project Alex designs hospital information system. Now he needs to write a component which puts a tentative diagnosis based on statistical data.

The program is loaded with information about N diseases and M possible symptoms. For each disease the chance that the patient appears to have each symptom is known. Similar information is known for healthy people. Alex assumes, that each of the patients has no more than one disease, and if disease has been cured (or doesn't exist), the various symptoms will occur independently.

The program loads information about K patients. For each particular patient it is known that some symptoms occur, and some do not, others have not been tested. We need to determine the possibility of this patient to be sick with every known disease.

Limit

$$1 \leq N \leq 200$$

$$1 \leq M \leq 200$$

$$1 \leq K \leq 200$$

Input

The first line contains two integers : N and M .

The next line contains M numbers — the probability that healthy person has corresponding symptom.

The following N lines contain information about diseases. Each of them contains $M + 1$ numbers: the probability that a person is sick with this disease (sum of the probabilities of all the diseases is less than 100%), and the probability of occurrence of each symptom, if a person is sick with the disease.

The next line contains a single integer K .

The following K lines contain information about patients: each of them contains a string M long, each of the symbols of which is +, if the patient appears to have the corresponding symptom, -, if it doesn't appear, or ?, if there is no detailed information about it.

All probabilities in the input file are specified in percentages, each value is a number from 0 to 100 with exactly two digits after the dot. It is guaranteed, that the combination of symptoms for each patient has a non-zero probability.

Output

Print the K strings — one for each patient. Each line should contain N numbers— the probability, that a patient has a corresponding disease. Print the probability as floating point number from 0 to 1 with an absolute percentage error not grater than 10^{-7} .

Example

stdin	stdout
1 2 10.00 10.00 40.00 90.00 90.00 3 ?? ++ +-	0.4000000 0.9818182 0.4000000
2 3 10.00 30.00 50.00 10.00 50.00 50.00 50.00 1.00 20.00 30.00 20.00 4 --? ++? ??- ---	0.0422797 0.0094707 0.4780115 0.0114723 0.0994036 0.0159046 0.0420408 0.0150674

Problem J. Zoological experiment

Input file: `stdin`
Output file: `stdout`
Time limit: 1 s
Memory limit: 256 Mb

Alex is a part of zoological project group and carries out experiments on fluffies. In one experiment, he puts two fluffies in the maze, that basically is an undirected graph. Every second each fluffy selects one of the vertices in the maze, adjacent to the current one, and jumps there. Fluffies select each of the adjacent vertices with equal probability. Although they are in the same maze, they do not interact with each other and move independently. At least, Alex thinks so. To test this hypothesis, he decided to measure how long, on average, a fluffy stays in the same vertex. To avoid errors, Alex averages fraction of a seconds, while fluffy stays in the vertex, for a long period. Alex also believes, that in spite of the fact that fluffies do not interact with each other, they keep the rhythm very precisely, and have been jumping all the time perfectly synchronous. He instructed you to make a theoretical calculations of this value for him.

Limit

$$2 \leq N \leq 100$$

$$1 \leq M \leq 10000$$

Input

The first line contains two integers: N and M — the number of vertices and edges in the maze.

Each of the next M lines contains two integers — vertex numbers (from 1 to N), connected by an edge. We assume that graph has no loops, parallel edges or isolated vertices.

The next line contains two integers from 1 to N — the numbers of vertices, where fluffies were originally placed. These numbers can match. Fluffies begin to move simultaneously.

Output

Print a single number — the amount of time the fluffy has to stay in the same vertex, in case that Alex's assumptions were correct.

Example

<code>stdin</code>	<code>stdout</code>
5 5 1 2 2 3 3 4 4 5 1 5 1 2	0.2

Problem K. A game (High)

Input file: `stdin`
Output file: `stdout`
Time limit: `1 sec`
Memory limit: `256 Mb`

Alex met a friend who invited him to play for the collectible card. The game divided on a rounds, and Bob has calculated that in every round he can win with probability $P\%$. If he wins the round, the friend gives him N_1 collectible cards, otherwise Bob gives his friend N_2 collectible cards. When the loser can have not enough card, the game ends, and the one who has run out of cards is considered to be defeated. Alex has M_1 collectible cards, and he knows that his friend has M_2 collectible cards. To decide whether or not to play, Bob wants to calculate the probability of his winnings in such circumstances, but he can not do it on his own. You need to help him.

Limit

$$1 \leq N_1, N_2, M_1, M_2 \leq 50$$

$$0 \leq P \leq 100$$

Input

The single line contains 5 integers: N_1 , N_2 , M_1 , M_2 and P .

Output

You should print one number — chances of Alex to win a whole game. Output response should be with an absolute percentage error of no more than 10^{-7} .

Example

<code>stdin</code>	<code>stdout</code>
1 1 10 10 50	0.5
2 1 10 10 40	0.8943674

Problem L. A dangerous game (High)

Input file: **stdin**
Output file: **stdout**
Time limit: 1 sec
Memory limit: 256 Mb

Alex got into serious trouble. He owes to the Mafia N dollars, and the time of reckoning has almost come. Alex is well known about the dangers of late return of the debt, so he wants to collect N dollars at any cost.

However, at the moment he has only M dollars. Alex's buddy, who learned about the problems suggested a game — but this time for money. Alex has K attempts to bet any amount from zero to the amount that he has a — not necessarily even a whole number. With probability of $P\%$ Alex gets bet double in volume, otherwise gets nothing. Alex does not care of how much money will remain, most importantly, to have N dollars to pay off. Alex, of course, trying to play good as he can. Find out what a chances that Alex will return his debt.

Limit

$$0 \leq N \leq 10^9$$

$$0 \leq M \leq 10^9$$

$$1 \leq K \leq 12$$

$$0 \leq P \leq 100$$

Input

The single line contains four integers: N , M , K and P .

Output

You should print a single number — probability to collect enough money. Print the answer with an absolute error of not more than 10^{-7} .

Example

stdin	stdout
1000000 750000 1 10	0.1
1000000 750000 2 10	0.19

Problem M. Hash-table (High)

Input file: `stdin`
Output file: `stdout`
Time limit: `1 sec`
Memory limit: `256 Mb`

Alex learned about a new data structure: a hash table with open addressing. Hash table with open addressing consists of N cells numbered from 1 to N . Each of them may be empty, or storing a value. When you insert a new value, it calculates the hash — random number between 1 to N (let's call it h) — after which, if the cell under number h is empty, the value is written into it. Otherwise, if the cell $h + 1$ is empty, the value is written into it, and otherwise it checks $h + 2$, $h + 3$ and so on. If the search has reached N -th cell, and it is not empty too, the search continues with the cell number 1. Thus, if there are an empty place somewhere, the value anyway will be added.

For example, if the value is equal to the hash h , the cell turned recorded $h + 2$ then we can assume, that there were two collisions occurred. Collisions slow down the hash table, so Alex decided to find out how many collisions will be in average, if in a empty hash table insert M different values. And calculate, as always, to you.

Limit

$$1 \leq N \leq 100$$

$$0 \leq M \leq N$$

Input

The single line contains 2 integers: N and M .

Output

You should print a single number — an average number of collisions. Print the answer with absolute or relative error of less than 10^{-7} .

Example

<code>stdin</code>	<code>stdout</code>
5 1	0.0
10 2	0.1