Florida International University
High School Programming Competition
Standard Division 1

A. Forced Choice
B. Watering the Fields
C. Rings
D. Pawn Shop
E. Kayaking Trip
F. Saint John Festival
G. Assigning Workstations
H. Tak and Cards
I. Iroha and the Grid
J. ETA
K. Just a Quiz

A. Forced Choice

Puff the Magic Dinosaur is a renowned magician. Due to the pandemic, he can only perform for his princess using the latest video conferencing tools. As a result, many of his usual in-person routines cannot be performed, and he must come up with a new trick.

Puff has decided to perform the following mentalist trick for the princess. First, Puff lays out N cards on the table. Each of these cards have a unique label *1,...,N*. Before the start of the show, Puff wrote down a prediction which is the label of one of the cards, put it into a sealed envelope, and mailed it to the princess.

During the show, Puff asks the princess to choose some of the remaining cards. Puff is careful to remind the princess to choose at least one but not all of the remaining cards. After the princess tells Puff which cards she chose, Puff would then say either "you chose to keep those cards" or "you chose to remove those cards". In the first case, the chosen cards are kept and all other cards are removed from the table. In the second case, the chosen cards are removed and the remaining cards are kept. This is repeated until there is only one card left. At this point Puff asks the princess to open the envelope, and magically the prediction matches the remaining card on the table.

Help Puff determine the appropriate response at each step.

Input

The first line of input contains three integers N $(2 \le N \le 200)$, which is the number of cards, P $(1 \le P \le N)$, which is the secret prediction, and S $(1 \le S \le N-1)$, which is the number of steps.

The next S lines describe the choices of the princess at each step. Each of these lines starts with the integer m $(1 \le m \le N-1)$, which is the number of cards chosen, followed by m distinct integers indicating the labels of the cards chosen. It is guaranteed that if Puff has carried out the previous steps correctly, the princess will only choose cards still on the table. Furthermore, the princess will always choose at least one but not all of the remaining cards.

It is guaranteed that there will be one card left after S steps if Puff has carried out all S steps correctly.

Output

For each step, display on a line either KEEP or REMOVE if Puff should keep the chosen cards or remove the chosen cards, respectively.

Sample Input 1	Sample Output 1
10 3 4 2 1 5 5 2 3 7 8 10 3 2 7 10 1 8	REMOVE KEEP REMOVE REMOVE

B. Watering the Fields

Due to a lack of rain, Farmer John wants to build an irrigation system to send water between his N fields ($1 \le N \le 2000$).

Each field i is described by a distinct point (x_i, y_i) in the 2D plane, with $0 \le x_i, y_i \le 1000$. The cost of building a water pipe between two fields i and j is equal to the squared Euclidean distance between them:

$(x_i - x_j)^2 + (y_i - y_j)^2$

FJ would like to build a minimum-cost system of pipes so that all of his fields are linked together -- so that water in any field can follow a sequence of pipes to reach any other field.

Unfortunately, the contractor who is helping FJ install his irrigation system refuses to install any pipe unless its cost (squared Euclidean length) is at least C ($1 \le C \le 1{,}000{,}000$).

Please help FJ compute the minimum amount he will need pay to connect all his fields with a network of pipes.

Input
Line 1: The integers N and C.

Lines 2..1+N: Line i+1 contains the integers x_i and y_i.

Output
Line 1: The minimum cost of a network of pipes connecting the fields, or -1 if no such network can be built.

Sample Input 1	Sample Output 1
3 11 0 2 5 0 4 3	46

Input Details:

There are 3 fields, at locations (0,2), (5,0), and (4,3). The contractor will only install pipes of cost at least 11.

Output Details:

FJ cannot build a pipe between the fields at (4,3) and (5,0), since its cost would be only 10. He therefore builds a pipe between (0,2) and (5,0) at cost 29, and a pipe between (0,2) and (4,3) at cost 17.

C. Rings

Dee Siduous is a botanist who specializes in trees. A lot of her research has to do with the formation of tree rings, and what they say about the growing conditions over the tree's lifetime. She has a certain theory and wants to run some simulations to see if it holds up to the evidence gathered in the field.

One thing that needs to be done is to determine the expected number of rings given the outline of a tree. Dee has decided to model a cross section of a tree on a two dimensional grid, with the interior of the tree represented by a closed polygon of grid squares. Given this set of squares, she assigns rings from the outer parts of the tree to the inner as follows: calling the non-tree grid squares "ring 0", each ring *n* is made up of all those grid squares that have at least one ring *(n−1)* square as a neighbor (where neighboring squares are those that share an edge). An example of this is shown in the figure below.

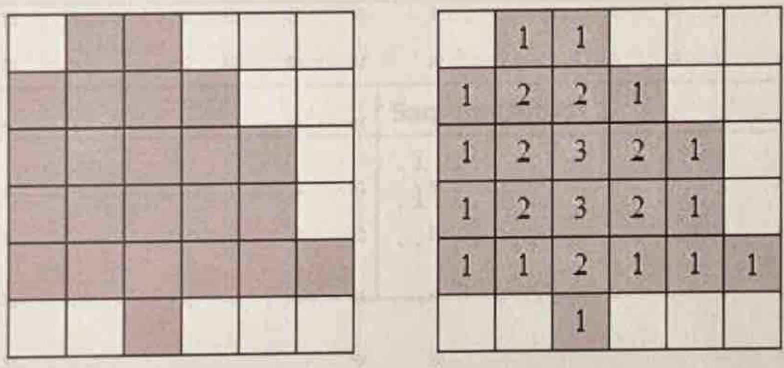


Figure D.1

Most of Dee's models have been drawn on graph paper, and she has come to you to write a program to do this automatically for her. This way she'll use less paper and save some ... well, you know.

Input

The input will start with a line containing two positive integers n m specifying the number of rows and columns in the tree grid, where *n,m≤100*. After this will be *n* rows containing *m* characters each. These characters will be either 'T' indicating a tree grid square, or

'.'

Output

Output a grid with the ring numbers. If the number of rings is less than 10, use two characters for each grid square; otherwise use three characters for each grid square. Right justify all ring numbers in the grid squares, and use '.' to fill in the remaining characters. If a row or column does not contain a ring number it should still be output, filled entirely with '.'s.

Sample Input 1	Sample Output 1
6 6 .TT... TTTT.. TTTTT. TTTTT. TTTTTT ..T...	...1.1...... .1.2.2.1.... .1.2.3.2.1.. .1.2.3.2.1.. .1.1.2.1.1.1 ....1.......

Sample Input 2	Sample Output 2
3 4 TT.. TT..	.1.1.... .1.1.... ....

D. Pawn Shop

You run a tight ship at the pawn shop. You arrange certain items in the window to be displayed to the street. You sometimes display the same type of item multiple times. For simplicity, we think of the items on display as a sequence of values where the value represents the type of item being shown.

For example, your display could be this sequence:

1 2 6 2 7 9 8 5

After coming back from your latest vacation, you find that your staff has completely rearranged the display by moving items around. Yikes! For example, the display above could be rearranged to:

2 6 1 2 9 7 5 8

You fear this could be the cause of confusion and may scare off repeat customers. But you don't have time to move items back to their original positions.

As a compromise, you will put dividers up in the window to partition the displayed items into groups of consecutive items. Each group should be a rearrangement of the types of items that were in those positions in your preferred arrangement.

More precisely, let $a_1,\ldots,a_N$ denote the first sequence and $b_1,\ldots,b_N$ denote the second sequence. You may place dividers around $i,i+1,\ldots,j$ if $b_i,b_{i+1},\ldots,b_j$ is a rearrangement of $a_i,a_{i+1},\ldots,a_j$. You do not need to put a divider at the beginning or end of the sequence. Note, if $a_i=b_i$ then a group may be formed using just this single index i.

With the sequences above, you could place dividers (shown here as #) at three positions as indicated here:

2 6 1 # 2 # 7 9 # 5 8

It is not possible to divide the sequence into more than four groups that have this property.

Given the two sequences, determine how to partition the new sequence into the maximum possible number of groups.

Input

The first line of input contains a single integer N ($1 \le N \le 300000$), which is the length of the two sequences.
The next line contains N integers $a_1,\dots,a_N$ ($1 \le a_i \le 10^9$), which is the original sequence.

The next line contains N integers $b_1,\dots,b_N$ ($1 \le b_i \le 10^9$), which is the rearranged sequence. The values $b_1,\dots,b_N$ are a rearrangement of the values $a_1,\dots,a_N$.

Output

Display the rearranged sequence with a valid and maximum placing of dividers (#).

If there are multiple possible solutions, display any of them.

Sample Input 1	Sample Output 1
8 1 2 6 2 7 9 8 5 2 6 1 2 9 7 5 8	2 6 1 # 2 # 9 7 # 5 8

Sample Input 2	Sample Output 2
4 1 1 2 1 2 1 1 1	2 1 1 # 1

Sample Input 3	Sample Output 3
3 1 2 5 2 5 1	2 5 1

E. Kayaking Trip

You are leading a kayaking trip with a mixed group of participants in the Stockholm archipelago, but as you are about to begin your final stretch back to the mainland you notice a storm on the horizon. You had better paddle as fast as you can to make sure you do not get trapped on one of the islands. Of course, you cannot leave anyone behind, so your speed will be determined by the slowest kayak. Time to start thinking; How should you distribute the participants among the kayaks to maximize your chance of reaching the mainland safely?

The kayaks are of different types and have different amounts of packing, so some are more easily paddled than others. This is captured by a speed factor c that you have already figured out for each kayak. The final speed v of a kayak, however, is also determined by the strengths s1 and s2 of the two people in the kayak, by the relation v=c(s1+s2). In your group you have some beginners with a kayaking strength of s_b, a number of normal participants with strength s_n and some quite experienced strong kayakers with strength s_e.

Input

The first line of input contains three non-negative integers b, n, and e, denoting the number of beginners, normal participants, and experienced kayakers, respectively. The total number of participants, b+n+e, will be even, at least 2, and no more than 100000. This is followed by a line with three integers s_b, s_n, and s_e, giving the strengths of the corresponding participants ($1 \le s_b < s_n < s_e \le 1000$). The third and final line contains $m=\frac{(b+n+e)}{2}$ integers $c_1,\ldots,c_m$ ($1 \le c_i \le 100000$ for each i), each giving the speed factor of one kayak.

Output

Output a single integer, the maximum speed that the slowest kayak can get by distributing the participants two in each kayak.

Sample Input 1	Sample output 1
3 1 0 40 60 90 18 20	1600

Sample Input 2	Sample Output 2
7 0 7 5 10 500 1 1 1 1 1 1 1	505

F. Saint John Festival

Porto's Festa de São João is one of Europe's liveliest street festivals. Its peak is the night of 23rd to 24th of June, with dancing parties from Ribeira to Foz all night long.

Time to celebrate, with friends, relatives, neighbours or simply with other people in streets, armed with colored plastic hammers, huge garlic flowers or a bunch of lemongrass to gently greet passers-by. Fireworks, grilled sardines, barbecues, bonfires, potted basil plants (*manjericos*) and the sky covered by incandescent sky lanterns (*balões de S.João*) launched from every corner make this party unique. The sky lanterns are made of thin paper and cannot be released until they are filled in with hot air. Sometimes they burn out still on ground or on the way up, if a sudden gust of wind catches them. For this reason, the successful launchers usually follow the movement of their sky lanterns, with a mixture of anxiety and joy, for as long as they can distinguish them in the sky.

We are not aware of any attempt to achieve a Guinness record of sky lanterns launched simultaneously (it would be a dreadful night for firemen if there were).

Can you imagine, thousands of people preparing their sky lanterns for release at the city park, within a region of larger ones that will be launched simultaneously?

The large sky lanterns can be used to identify their positions in the sky afterwards, in order to count the surviving ones at an observation instant.

Task

Given the positions of the large sky lanterns and the positions of the small ones, determine the number of small sky lanterns that are in the interior or on the boundary of some triangle defined by any three of the large ones.

Input

The first line has an integer L that defines the number of the large sky lanterns at the observation instant. Each of the following L lines contains a pair of integers separated by a space that gives the coordinates (x,y) of a large sky lantern. After that, there is a line with an integer S that defines the number of small sky lanterns and S lines, each defining the position of a small sky lantern. The height is irrelevant for us. All the given points are distinct and there are at least three points representing large sky lanterns that are not collinear.

Output

The output has a single line with the number of small sky lanterns that are in the interior or on the boundary of some triangle defined by any three of the large sky lanterns.

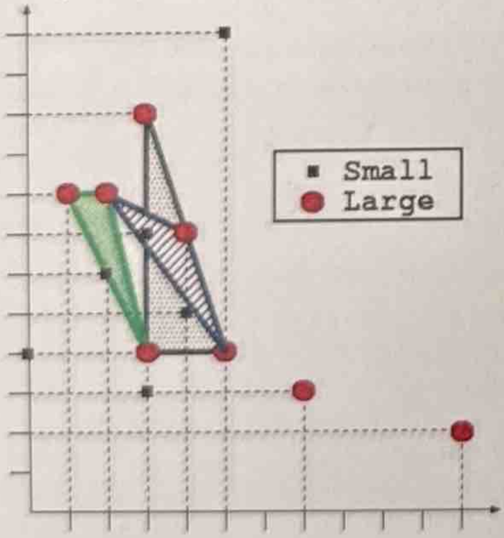
Constraints

$3 \leq L \leq 10000$ Number of large sky lanterns.

$1 \leq S \leq 50000$ Number of small sky lanterns.

$0 \leq x, y \leq 2^{30}$ Bounds for coordinates.

Sample Input Illustration



Sample Input 1	Sample Output 1
8 3 4 2 8 5 4 1 8 4 7 3 10 11 2 7 3 6 5 12 3 7 3 3 4 5 0 4 2 6	3

G. Assigning Workstations

Penelope is part of the admin team of the newly built supercomputer. Her job is to assign workstations to the researchers who come here to run their computations at the supercomputer.

Penelope is very lazy and hates unlocking machines for the arriving researchers. She can unlock the machines remotely from her desk, but does not feel that this menial task matches her qualifications. Should she decide to ignore the security guidelines she could simply ask the researchers not to lock their workstations when they leave, and then assign new researchers to workstations that are not used any more but that are still unlocked. That way, she only needs to unlock each workstation for the first researcher using it, which would be a huge improvement for Penelope.

Unfortunately, unused workstations lock themselves automatically if they are unused for more than m minutes. After a workstation has locked itself, Penelope has to unlock it again for the next researcher using it. Given the exact schedule of arriving and leaving researchers, can you tell Penelope how many unlockings she may save by asking the researchers not to lock their workstations when they leave and assigning arriving researchers to workstations in an optimal way? You may assume that there are always enough workstations available.

Input

The input consists of:

one line with two integers n ($1 \le n \le 300000$), the number of researchers, and m ($1 \le m \le 10^8$), the number of minutes of inactivity after which a workstation locks itself;

n lines each with two integers a and s ($1 \le a,s \le 10^8$), representing a researcher that arrives after a minutes and stays for exactly s minutes.

Output

Output the maximum number of unlockings Penelope may save herself.

Sample Input 1	Sample Output 1
3 5 1 5 6 3 14 6	2

Sample Input 2	Sample Output 2
5 10 2 6 1 2 17 7 3 9 15 6	3

H. Tak and Cards

Tak has N cards. On the i-th $(1 \leq I \leq N)$ card is written an integer x_i. He is selecting one or more cards from these N cards, so that the average of the integers written on the selected cards is exactly A. In how many ways can he make his selection?

Constraints

$1 \leq N \leq 50$
$1 \leq A \leq 50$
$1 \leq x_i \leq 50$
N,A,x_i are integers.

Input

The input is given from Standard Input in the following format:

```
N A x1 x2
x3 .... xn
```

Output

Print the number of ways to select cards such that the average of the written integers is exactly A.

Sample Input 1	Sample Output 1
4 8 7 9 8 9	5

- **Explanation**

  The following are the *5* ways to select cards such that the average is *8*:
  - Select the *3*-rd card.
  - Select the *1*-st and *2*-nd cards.
  - Select the *1*-st and *4*-th cards.
  - Select the *1*-st, *2*-nd and *3*-rd cards.
  - Select the *1*-st, *3*-rd and *4*-th cards.

Sample Input 2	Sample Output 2
3 8 6 6 9	0

Sample Input 3	Sample Output 3
8 5 3 6 2 8 7 6 5 9	19

Sample Input 4	Sample Output 4
33 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3	8589934591

- The answer may not be fit into 32-bit integer

I. Iroha and the Grid

We have a large square grid with H rows and W columns. Iroha is now standing in the top-left cell. She will repeat going right or down to the adjacent cell, until she reaches the bottom-right cell.

However, she cannot enter the cells in the intersection of the bottom A rows and the leftmost B columns. (That is, there are $A \times B$ forbidden cells.) There is no restriction on entering the other cells.

Find the number of ways she can travel to the bottom-right cell.

Since this number can be extremely large, print the number modulo 10^9+7.

Constraints

$1 \leqq H,W \leqq 100{,}000$

$1 \leqq A < H$

$1 \leqq B < W$

Input

The input is given from Standard Input in the following format:

```
H W A B
```

Output

Print the number of ways she can travel to the bottom-right cell, modulo 10^9+7.

Sample Input 1	Sample Output 1
2 3 1 1	2

Explanation

We have a 2×3 grid, but entering the bottom-left cell is forbidden. The number of ways to travel is two: "Right, Right, Down" and "Right, Down, Right".

Sample Input 2	Sample Output 2
`10 7 3 4`	`3570`

There are 12 forbidden cells.

Sample Input 3	Sample Output 3
`100000 100000 99999 99999`	`1`

Sample Input 4	Sample Output 4
`100000 100000 44444 55555`	`738162020`

J. ETA

You want to design a level for a computer game. The level can be described as a connected undirected graph with vertices numbered from 1 to n. In the game, the player's character is dropped at one of the n vertices uniformly at random and their goal is to reach the exit located at vertex 1 as quickly as possible. Traversing an edge takes exactly 1 second.

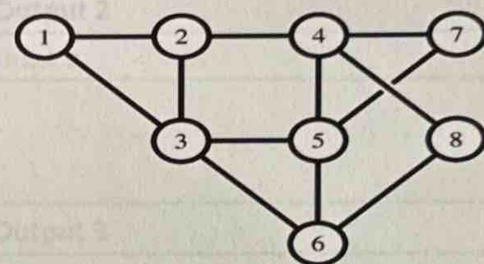


Figure 1: Illustration of Sample Output 3, a level where the average optimal time to reach vertex 1 is 74.

The difficulty of the level is determined by the average optimal time to reach the exit. Given a target value for this average optimal time, construct a level so that this target value is reached. See Figure 1 for an example.

Input

The input consists of:

1. One line with two coprime integers *a* and *b* ($1 \le a, b \le 1000$) separated by a '/', giving the desired average optimal time to reach the exit as the fraction *a/b*.

Output

If no connected graph with the average optimal time *a/b* to reach vertex 1 exists, output "impossible".

Otherwise, output one such graph in the following format:

1. Two integers *n and m* ($1 \le n, m \le 10^6$), the number of vertices and the number of edges.
2. *m* pairs of integers *u and v* ($1 \le u, v \le n$), indicating an edge between vertices *u and v*. The graph may include self loops and parallel edges. You are given that if there exists a valid graph, then there also exists one with $1 \le n, m \le 10^6$.

If there are multiple valid solutions, you may output any one of them.

Sample Input 1	Sample Output 1
1/2	2 1 1 2

Sample Input 2	Sample Output 2
1/3	impossible

Sample Input 3	Sample Output 3
7/4	8 12 1 2 1 3 2 3 2 4 3 5 3 6 4 5 5 6 4 7 5 7 4 8 6 8

K. Just a Quiz

In the TV quiz *Monstermind*, a contestant chooses a topic and is then asked questions about it during a fixed period of time. The contestant earns one point for each correct answer. When the time runs out, the contestant must be silent.

Teresa has figured out such a niche topic that she knows all possible questions that may be asked about it, as well as all the answers. Since the competition is fierce, she has decided to sometimes answer a question before the host finishes reading it. The host picks each question uniformly at random from the pool of possible questions, and each question may be asked multiple times. When reading a question, the host reads at a pace of one word per second.

Teresa can interrupt the host mid-question—between words, or even before hearing the first word— but not mid-word—that would be extremely impolite. Answering also takes one second, and the host will start reading another question immediately after an answer—unless Teresa interrupts again.

She wrote a program to help her choose the best moment to answer, and now there is only one question left for you. How many points does she expect to score?

For example, in the first sample test case the answer is completely determined after hearing one word, so it is optimal to answer after hearing it, and Teresa answers 2 questions correctly in 4 seconds. In the second sample test case, if the first word is What, then it takes too much time to wait for the question to finish. Therefore Teresa says Now! 4 times and expects to get 1/3 of the answers right.

Input

The first line contains two integers *t and n* ($1 \le t \le 100$, $1 \le n \le 100\ 000$), the duration of the quiz and the number of questions. Each of the following n lines contains a question, which is a space-separated list of words terminated by a question mark; and an answer, which is a single word.

Each word is a sequence of non-space ASCII printable characters, between the ASCII values of '!' and '˜'. Only the last word of a question has a question mark ('?'). You can assume that no question is a prefix of another and that punctuation marks are part of a word. Words spelled with different upper/lower case are assumed to be different.

It is guaranteed that the total number of word characters is at most 100 000.

Output

Output the expected score of an optimal strategy. Answers within a relative or absolute error of 10^{-6} will be accepted.

Sample Input 1	Sample Output 1
4 4 How much is 6 times 9? 42 How much is 9 times 6? 42 Is there intelligent life on Earth? Probably What is the air speed velocity of an unladen swallow? African?	2.0000000000

Sample Input 2	Sample Output 2
4 3 What do we send? Code What do we want? Accepted When do we want it? Now!	1.333333333