Problem A. Broken Audio Signal

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Nathan O. Davis is a student at the department of integrated systems. Today's agenda in the class is audio signal processing. Nathan was given a lot of homework out. One of the homework was to write a program to process an audio signal. He copied the given audio signal to his USB memory and brought it back to his home.

When he started his homework, he unfortunately dropped the USB memory to the floor. He checked the contents of the USB memory and found that the audio signal data got broken. There are several characteristics in the audio signal that he copied.

- The audio signal is a sequence of N samples.
- Each sample in the audio signal is numbered from 1 to N and represented as an integer value.
- Each value of the odd-numbered sample(s) is strictly smaller than the value(s) of its neighboring sample(s).
- Each value of the even-numbered sample(s) is strictly larger than the value(s) of its neigh- boring sample(s).

He got into a panic and asked you for a help. You tried to recover the audio signal from his USB memory but some samples of the audio signal are broken and could not be recovered. Fortunately, you found from the metadata that all the broken samples have the same integer value.

Your task is to write a program, which takes the broken audio signal extracted from his USB memory as its input, to detect whether the audio signal can be recovered uniquely.

Input

The first line of the input consists of an integer, N ($2 \leq N \leq 1,000$). N denotes the number of samples in the given audio signal. The second line of each dataset consists of N values separated by spaces. The i-th value, a_i , is either a character 'x' or an integer between -10^9 and 10^9, inclusive. It represents the i-th sample of the broken audio signal. If a_i is a character 'x', it denotes that i-th sample in the audio signal is broken. Otherwise it denotes the value of the i-th sample.

Output

Output the value of the broken samples in one line if the original audio signal can be recovered uniquely. If there are multiple possible values, output "`ambiguous`". If there are no possible values, output "`none`".

Examples

standard input	standard output
5 1 x 2 4 x	3
2 x x	none
2 1 2	ambiguous
2 2 1	none
2 1000000000 x	ambiguous
4 x 2 1 x	none

Problem B. Restore Calculation

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

The Animal School is a primary school for animal children. You are a fox attending this school. One day, you are given a problem called "Arithmetical Restorations" from the rabbit teacher, Hanako. Arithmetical Restorations are the problems like the following:

- You are given three positive integers, A, B and C.
- Several digits in these numbers have been erased.
- You should assign a digit in each blank position so that the numbers satisfy the formula $A+B=C$.
- The first digit of each number must not be zero. It is also the same for single-digit numbers.

You are clever in mathematics, so you immediately solved this problem. Furthermore, you decided to think of a more difficult problem, to calculate the number of possible assignments to the given Arithmetical Restorations problem. If you can solve this difficult problem, you will get a good grade. Shortly after beginning the new task, you noticed that there may be too many possible assign- ments to enumerate by hand. So, being the best programmer in the school as well, you are now trying to write a program to count the number of possible assignments to Arithmetical Restoration problems.

Input

Input consists of three strings, A, B and C. They indicate that the sum of A and B should be C. Each string consists of digits (`0-9`) and/or question mark (`?`). A question mark (`?`) indicates an erased digit. You may assume that the first character of each string is not `0`, and each dataset has at least one `?`. It is guaranteed that each string contains between 1 and 50 characters, inclusive. You can also assume that the lengths of three strings are equal.

Output

Output the number of possible assignments to the given problem modulo $10^9 + 7$. Note that there may be no way to solve the given problems because Ms. Hanako is a careless rabbit.

Example

standard input	standard output
3?4 12? 5?6	2
?2?4 5?7? ?9?2	40
????? ????? ?????	200039979

Note

The answer of the first test is 2. They are shown below.

- $384 + 122 = 506$
- $394 + 122 = 516$

Problem C. SIRO Challenge

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 mebibytes

You are now participating in the Summer Training Camp for Programming Contests with your friend Jiro, who is an enthusiast of the ramen chain SIRO. Since every SIRO restaurant has its own tasteful ramen, he wants to try them at as many different restaurants as possible in the night. He doesn't have plenty of time tonight, however, because he has to get up early in the morning tomorrow to join a training session. So he asked you to find the maximum number of different restaurants to which he would be able to go to eat ramen in the limited time.

There are n railway stations in the city, which are numbered 1 through n. The station s is the nearest to the camp venue. m pairs of stations are directly connected by the railway: you can move between the stations a_i and b_i in c_i minutes in the both directions. Among the stations, there are l stations where a SIRO restaurant is located nearby. There is at most one SIRO restaurant around each of the stations, and there are no restaurants near the station s. It takes e_i minutes for Jiro to eat ramen at the restaurant near the station j_i .

It takes only a negligibly short time to go back and forth between a station and its nearby SIRO restaurant. You can also assume that Jiro doesn't have to wait for the ramen to be served in the restaurants.

Jiro is now at the station s and have to come back to the station in t minutes. How many different SIRO's can he taste?

Input

The first line of the input contains five integers: n for the number of stations, m for the number of directly connected pairs of stations, l for the number of SIRO restaurants, s for the starting-point station, and t for the time limit for Jiro ($2 \le n \le 300$, $1 \le m \le 5000$, $1 \le l \le 16$, $1 \le s \le n$, $1 \le t \le 10^5$). Each of the following m lines contains three integers: a_i and b_i for the connected stations, and c_i for the time it takes to move between the two stations ($1 \le a_i, b_i \le n$, $1 \le c_i \le 1000$, $a_i \ne b_i$, $(a_i, b_i) \ne (a_j, b_j)$ and $(a_i, b_i) \ne (b_j, a_j)$ for any $i \ne j$). Each of the following l lines contains two integers: j_i for the station where a SIRO restaurant is located, and e_i for the time it takes for Jiro to eat at the restaurant ($1 \le j_i \le n$, $1 \le e_i \le 15$, $j_i \ne s$, all j_i are pairwise distinct). Note that there may be some stations not reachable from the starting point s.

Output

Output the maximum number of different restaurants where Jiro can go within the time limit.

Example

standard input	standard output
2 1 1 1 10	1
1 2 3	
2 4	
2 1 1 1 9	0
1 2 3	
2 4	
4 2 2 4 50	1
1 2 5	
3 4 5	
2 15	
3 15	
4 6 3 1 29	3
1 2 20	
3 2 10	
4 1 5	
3 1 5	
2 4 3	
3 4 4	
2 1	
4 5	
3 3	

Problem D. Everlasting -One-

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 mebibytes

"Everlasting -One-" is an award-winning online game launched this year. This game has rapidly become famous for its large number of characters you can play.

In this game, a character is characterized by attributes. There are N attributes in this game, numbered 1 through N. Each attribute takes one of the two states, light or darkness. It means there are 2^N kinds of characters in this game.

You can change your character by job change. Although this is the only way to change your character's attributes, it is allowed to change jobs as many times as you want. The rule of job change is a bit complex. It is possible to change a character from A to B if and only if there exist two attributes a and b such that they satisfy the following four conditions:

- The state of attribute a of character A is light.
- The state of attribute b of character B is light.
- There exists no attribute c such that both characters A and B have the light state of attribute c.
- A pair of attribute (a, b) is compatible.

Here, we say a pair of attribute (a, b) is compatible if there exists a sequence of attributes $c_1, c_2, \ldots, c_n$ satisfying the following three conditions:

- $c_1 = a$.
- $c_n = b$.
- Either (c_i, c_{i+1}) or (c_{i+1}, c_i) is a special pair for all $i = 1, 2, \ldots, n-1$.

You will be given the list of special pairs.

Since you love this game with enthusiasm, you are trying to play the game with all characters (it's really crazy). However, you have immediately noticed that one character can be changed to a limited set of characters with this game's job change rule. We say character A and B are essentially different if you cannot change character A into character B by repeating job changes. Then, the following natural question arises; how many essentially different characters are there? Since the output may be very large, you should calculate the answer modulo $10^9 + 7$.

Input

The first line of the input contains two integers N and M ($1 \leq N \leq 10^5$ and $0 \leq M \leq 10^5$). Then M lines follow. The i-th line contains two integers a_i and b_i ($1 \leq a_i < b_i \leq N$) which denote the i-th special pair.

It is guaranteed that $(a_i, b_i) \neq (a_j, b_j)$ if $i \neq j$ and total size of input is less than 1.2 MiB.

Output

Print the number of essentially different characters modulo $10^9 + 7$.

Example

standard input	standard output

3 2
1 2
2 3
	3

5 0	32
100000 0	607723520

Problem E. Putter

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 mebibytes

Fox Ciel is practicing miniature golf, a golf game played with a putter club only. For improving golf skills, she believes it is important how well she bounces the ball against walls.

The field of miniature golf is in a two-dimensional plane and surrounded by N walls forming a convex polygon. At first, the ball is placed at (s_x, s_y) inside the field. The ball is small enough to be regarded as a point.

Ciel can shoot the ball to any direction and stop the ball whenever she wants. The ball will move in a straight line. When the ball hits the wall, it rebounds like mirror reflection (i.e. incidence angle equals reflection angle).

For practice, Ciel decided to make a single shot under the following conditions:

- The ball hits each wall of the field exactly once.
- The ball does NOT hit the corner of the field.

Count the number of possible orders in which the ball hits the walls.

Input

The first line contains an integer N ($3 \leq N \leq 8$). The next line contains two integers s_x and s_y ($-50 \leq s_x, s_y \leq 50$), which describe the coordinates of the initial position of the ball. Each of the following N lines contains two integers x_i and y_i ($-50 \leq x_i, y_i \leq 50$), which describe the coordinates of each corner of the field. The corners are given in counterclockwise order. You may assume given initial position (s_x, s_y) is inside the field and the field is convex.

It is guaranteed that there exists a shoot direction for each valid order of the walls that satisfies the following condition: distance between the ball and the corners of the field (x_i, y_i) is always greater than 10^{-6} until the ball hits the last wall.

Output

Print the number of valid orders of the walls in a line.

Example

standard input	standard output

4
0 0
-10 -10
10 -10
10 10
-10 10
	8

Problem F. Shipura

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes
Feedback:	

Dr. Suposupo developed a programming language called Shipura. Shipura supports only one binary operator "`»`" and only one unary function "`S < >`".

$x >> y$ is evaluated to the greatest integer not exceeding $x/2y$, and $S < x >$ is evaluated to $x^2 \bmod 10^9 + 7$ (that is, the remainder when x^2 is divided by $10^9 + 7$).

The operator "`»`" is left-associative. For example, the expression "`x»y»z`" is interpreted as "`(x»y)»z`", not as "`x»(y»z)`". Note that these parentheses do not appear in actual Shipura expressions.

The syntax of Shipura is given (in BNF; Backus-Naur Form) as follows:

```
expr    ::=   term | expr sp ">>" sp term
term    ::=   number | "S" sp "<" sp expr sp ">"
sp      ::=   "" | sp " "
number  ::=   digit | number digit
digit   ::=   "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

The start symbol of this syntax is expr that represents an expression in Shipura. In addition, number is an integer between 0 and 10^9 inclusive, written without extra leading zeros. Write a program to evaluate Shipura expressions.

Input

The input is represented by a line which contains a valid expression in Shipura. You can assume the total size of the input file does not exceed $10^5 + 7$ bytes.

Output

Output a line containing the evaluated value of the expression.

Example

standard input	standard output
`S< S< 12 » 2 > >`	81
`123 » 1 » 1`	30
`1000000000   »129`	0
`S<S<S<S<S<2»»>`	294967268
`S  <S< S<2013    »> 11 »> 10 >`	14592400

Problem G. Floating Islands

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1.5 seconds
Memory limit:	512 mebibytes

You have just arrived in a small country. Unfortunately a huge hurricane swept across the country a few days ago.

The country is made up of n islands, numbered 1 through n. Many bridges connected the islands, but all the bridges were washed away by a flood. People in the islands need new bridges to travel among the islands again.

The problem is cost. The country is not very wealthy. The government has to keep spending down. They asked you, a great programmer, to calculate the minimum cost to rebuild bridges. Write a program to calculate it!

Each bridge connects two islands bidirectionally. Each island i has two parameters d_i and p_i. An island i can have at most d_i bridges connected. The cost to build a bridge between an island i and another island j is calculated by $|p_i - p_j|$. Note that it may be impossible to rebuild new bridges within given limitations although people need to travel between any pair of islands over (a sequence of) bridges.

Input

Everything in the input is an integer. n ($2 \le n \le 4000$) on the first line indicates the number of islands. Then n lines follow, which contain the parameters of the islands. p_i ($1 \le p_i \le 10^9$) and d_i ($1 \le d_i \le n$) denote the parameters of the island i.

Output

Output the minimum cost in a line if it is possible to rebuild bridges within given limitations. Otherwise, output -1 in a line.

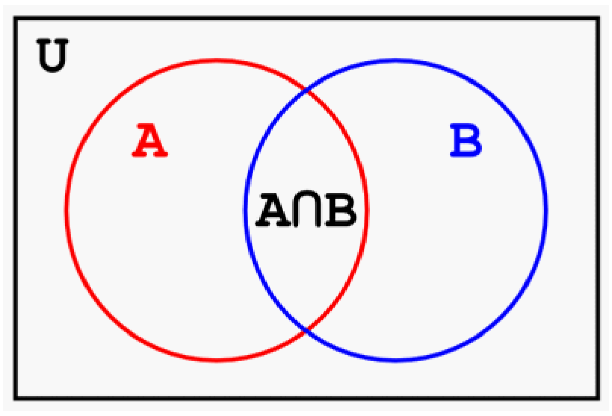
Example

standard input	standard output
4	18
1 1	
8 2	
9 1	
14 2	
4	634
181 4	
815 4	
634 4	
370 4	
4	-1
52 1	
40 1	
81 2	
73 1	
10	916
330 1	
665 3	
260 1	
287 2	
196 3	
243 1	
815 1	
287 3	
330 1	
473 4	

Problem H. Venn Diagram

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	512 mebibytes

Alice is a private teacher. One of her job is to prepare the learning materials for her student. Now, as part of the materials, she is drawing a Venn diagram between two sets A and B. Venn diagram is a diagram which illustrates the relationships among one or more sets. For example, a Venn diagram between two sets A and B is drawn as illustrated below. The rectangle corresponds to the universal set U. The two circles in the rectangle correspond to two sets A and B, respectively. The intersection of the two circles corresponds to the intersection of the two sets, i.e. $A \cap B$.



Alice, the mathematics personified, came up with a special condition to make her Venn diagram more beautiful. Her condition is that the area of each part of her Venn diagram is equal to the number of elements in its corresponding set. In other words, one circle must have the area equal to $|A|$, the other circle must have the area equal to $|B|$, and their intersection must have the area equal to $|A \cap B|$. Here, $|X|$ denotes the number of elements in a set X.

Alice already drew a rectangle, but has been having a trouble figuring out where to draw the rest, two circles, because she cannot even stand with a small error human would make. As an old friend of Alice's, your task is to help her by writing a program to determine the centers and radii of two circles so that they satisfy the above condition.

Input

The first two integers U_W and U_H ($1 \le U_W, U_H \le 100$) denote the width and height of the rectangle which corresponds to the universal set U, respectively. The next three integers $|A|$, $|B|$ and $|A \cap B|$ ($1 \le |A|, |B| \le 10^4$ and $0 \le |A \cap B| \le min(|A|, |B|)$) denote the numbers of elements of the set A, B and $A \cap B$, respectively.

You may assume that, even if U_W and U_H would vary within plus/minus 0.01, it would not change whether you can draw two circles under the Alice's condition.

Output

Print the centers and radii of the two circles that satisfy the Alice's condition as follows: X_A and Y_A are the coordinates of the center of the circle which corresponds to the set A. R_A is the radius of the circle which corresponds to the set A. X_B, Y_B and R_B are the values for the set B. These values must be separated by a space. If it is impossible to satisfy the condition, output "`impossible`"

The area of each part must not have an absolute error greater than 0.0001. Also, the two circles must stay inside the rectangle with a margin of 0.0001 for error, or more precisely, all of the following four conditions must be satisfied: $X_A - RA \geq -0.0001$, $X_A + R_A \leq U_W + 0.0001$, $Y_A - R_A/ge - 0.0001$, $Y_A + R_A \leq U_H + 0.0001$. The same conditions must hold for X_B, Y_B and R_B.

Example

standard input	standard output
`10 5 1 1 0`	`1 1 0.564189584 3 1 0.564189584`
`10 5 2 2 1`	`1 1 0.797884561 1.644647246 1 0.797884561`
`10 10 70 70 20`	`impossible`

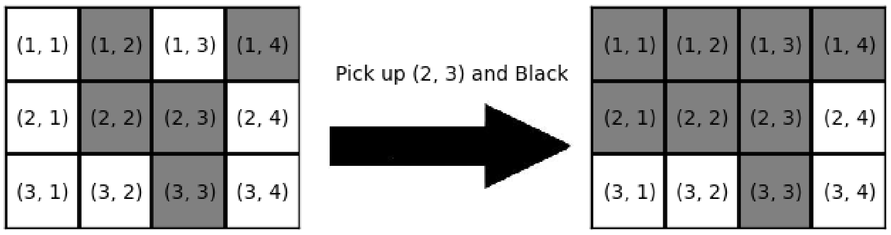
Problem I. Overwriting Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 mebibytes

You have a rectangular board with square cells arranged in H rows and W columns. The rows are numbered 1 through H from top to bottom, and the columns are numbered 1 through W from left to right. The cell at the row i and the column j is denoted by (i, j). Each cell on the board is colored in either Black or White. You will paint the board as follows:

1. Choose a cell (i, j) and a color c, each uniformly at random, where $1 \le i \le H$, $1 \le j \le W$, and $c \in \{Black, White\}$.
2. Paint the cells (i', j') with the color c for any $1 \le i' \le i$ and $1 \le j' \leqslant j$.

Here's an example of the painting operation. You have a 3×4 board with the coloring depicted in the left side of the figure below. If your random choice is the cell $(2, 3)$ and the color Black, the board will become as shown in the right side of the figure. 6 cells will be painted with Black as the result of this operation. Note that we count the cells "painted" even if the color is not actually changed by the operation, like the cell $(1, 2)$ in this example.



Given the initial coloring of the board and the desired coloring, you are supposed to perform the painting operations repeatedly until the board turns into the desired coloring. Write a program to calculate the expected total number of painted cells in the sequence of operations.

Input

The first line of the input contains two integers H and W $(1 \le H, W \le 5)$, the numbers of rows and columns of the board respectively. Then given are two coloring configurations of the board, where the former is the initial coloring and the latter is the desired coloring. A coloring configuration is described in H lines, each of which consists of W characters. Each character is either B or W, denoting a Black cell or a White cell, respectively. There is one blank line between two configurations. You can assume that the resulting expected value will not exceed 10^9.

Output

Your program should output the expected value in a line. The absolute error or the relative error in your answer must be less than 10^{-6}.

Example

standard input	standard output
1 2 BB WW	6.0000000000
2 1 B W B W	0.0000000000
2 2 BW BW WW WW	12.8571428571
3 4 BBBB BBBB BBBB WWWW WWWW WWWW	120.0000000000
5 5 BBBBB BBBBB BBBBB BBBBB BBBBB BBBBB BBBWB BBBBB BWBBB BBBBB	23795493.8449918639

Problem J. Magical Switches

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	512 mebibytes

You are given a rectangular board divided into square cells. The number of rows and columns in this board are 3 and $3M + 1$, respectively, where M is a positive integer. The rows are numbered 1 through 3 from top to bottom, and the columns are numbered 1 through $3M + 1$ from left to right. The cell at the i-th row and the j-th column is denoted by (i, j).

Each cell is either a floor cell or a wall cell. In addition, cells in columns $2, 3, 5, 6, \ldots, 3M-1, 3M$ (numbers of form $3k - 1$ or $3k$, for $k = 1, 2, \ldots, M$) are painted in some color. There are 26 colors which can be used for painting, and they are numbered 1 through 26. The other cells (in columns $1, 4, 7, \ldots, 3M + 1$) are not painted and each of them is a floor cell.

You are going to play the following game. First, you put a token at cell $(2, 1)$. Then, you repeatedly move it to an adjacent floor cell. Two cells are considered adjacent if they share an edge. It is forbidden to move the token to a wall cell or out of the board. The objective of this game is to move the token to cell $(2, 3M + 1)$.

For this game, 26 magical switches are available for you! Switches are numbered 1 through 26 and each switch corresponds to the color with the same number. When you push switch x, each floor cell painted in color x becomes a wall cell and each wall cell painted in color x becomes a floor cell, simultaneously.

You are allowed to push some of the magical switches ONLY BEFORE you start moving the token. Determine whether there exists a set of switches to push such that you can achieve the objective of the game, and if there does, find such a set.

Input

The input begins with a line containing an integer M ($1 \le M \le 1,000$). The following three lines, each containing $3M + 1$ characters, represent the board. The j-th character in the i-th of those lines describes the information of cell (i, j), as follows:

- The x-th uppercase letter indicates that cell (i, j) is painted in color x and it is initially a floor cell.
- The x-th lowercase letter indicates that cell (i, j) is painted in color x and it is initially a wall cell.
- A period (‘`.`’) indicates that cell (i, j) is not painted and so it is a floor cell. Here you can assume that j will be one of $1, 4, 7, \ldots, 3M + 1$ if and only if that character is a period. The end of the input is indicated by a line with a single zero.

Output

Output -1 if you cannot achieve the objective. Otherwise, output the set of switches you push in order to achieve the objective. First print n —- the number of switches you push and then print $s_1, s_2, \ldots, s_n$ are uppercase letters corresponding to switches you push, where the x-th uppercase letter denotes switch x. These uppercase letters $s_1, s_2, \ldots, s_n$ must be distinct, while the order of them does not matter. Note that it is allowed to output $n = 0$ (with no following uppercase letters) if you do not have to push any switches. See the sample output for clarification. If there are multiple solutions, output any one of them.

Example

standard input

```
3
.aa.cA.Cc.
.bb.Bb.AC.
.cc.ac.Ab.
```

standard output

```
3 B C E
```

standard input

```
1
.Xx.
.Yy.
.Zz.
```

standard output

```
-1
```

standard input

```
6
.Aj.fA.aW.zA.Jf.Gz.
.gW.GW.Fw.ZJ.AG.JW.
.bZ.jZ.Ga.Fj.gF.Za.
```

standard output

```
3 J A G
```

standard input

```
9
.ab.gh.mn.st.yz.EF.KL.QR.WA.
.cd.ij.op.uv.AB.GH.MN.ST.XB.
.ef.kl.qr.wx.CD.IJ.OP.UV.yz.
```

standard output

```
10 A B G H M N S T Y Z
```

standard input

```
2
.AC.Mo.
.IC.PC.
.oA.CM.
```

standard output

```
0
```

standard input

```
20
.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.qb.
.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.qb.
.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.QB.qb.
```

standard output

```
2 Q B
```