Problem A. Artifact Autentification

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

This is an interactive problem.

In the famous video game ¡¡Heroes of Debug and Submit¿¿, the main hero finds himself in some kind of maze structure and has to find a way out. Each level is constructed with several caves, platforms, rooms or other kind of interesting spots that are connected with paths of different types. In each spot, there is a single distinguishable and unique artefact that can be activated by the hero. To escape from one level, the player has to activate the correct artefacts in the correct order. Riddles hidden inside the maze help him to solve the level.

Byteazar obtained a map of a single level and its solution from the internet. He only needs to find out whether this map represents the current level he is in.

While wandering around from spot to spot, the player can see the artefact in his current spot as well as all paths leading away from it. The game implements 26 different types of paths such as paved roads ('P'), bridges ('B') or dirt roads ('D'). In each spot all paths leading away are of different types.

Byteazar does not know his initial position but he can always decide which path to follow. For each spot he visit, he is provided with the name of the artifact there and the types of paths leading away.

Byteazar asked you to find out whether this level matches the one on his map.

Interaction Protocol

The jury program sends to your program the description of the given map: first line of the input contains one integer n ($2 \le n \le 1\,000$), the number of spots on the map. i-th of following n lines, contains an integer k followed by k pairs $t_1\ m_1, \ldots, t_k\ m_k$, describing the paths leading away from spot i. Each path is given by its type t_j (t_j is an uppercase letter) and destination m_j ($1 \le m_j \le n, m_j \ne i$).

All paths are bidirectional and appear twice in the input, once for each side. You can safely assume that all spots are reachable from any position in the maze.

Then, the game proceeds in turns. In each turn, the jury prigram sends you the description of your current spot: a line with a lowercase string a ($1 \le |a| \le 20$) and an uppercase string s ($1 \le |s| \le 26$), the name of the artifact you see at this spot and the description of the paths leading away. No two spots will have the same artefact. All letters in s are distinct and uppercase with no particular order. The order can vary when entering the same spot again.

Your submission must respond with one of the following:

- `W` c – you want to walk along the path of type c, where c is a valid uppercase character.
- `R` i – you are sure the level corresponds to the map. The integer i ($1 \le i \le n$) should be the number of the spot on the initial map that you believe you are currently in.
- `R ambiguous` – the map matches the level, but the matching is not unique.
- `R no` – the level does not correspond to your map.

After every valid response starting with 'W', the jury program gives you the description of the new spot you are currently in. You are not allowed to do more than 250 000 walks. After a response starting with 'R', the game ends. When you send an invalid response, the game ends and your submission will be judged as "Wrong Answer". After each command you send, you should `flush` the standard output to ensure that it is sent to the game. For example, you can use `fflush(stdout)` in C++, `System.out.flush()` in Java, and `sys.stdout.flush()` in Python.

Examples

standard input	standard output
3 2 D 2 B 3 2 P 3 D 1 2 P 2 B 1 source DB monument PD portal PB source DB	 W D W P W B R 1
3 2 D 2 B 3 2 P 3 D 1 2 P 2 B 1 shrine PD arena LD	 W D R no
2 2 P 2 D 2 2 D 1 P 1 temple PD arena PD temple PD	 W P W D R ambiguous

Note

There is overview of first sample:

In the example interactions above, the output of the judge program is on the left and one possible correct output of a submission is on the right.

The empty lines on both sides only serve to emphasise the chronological order of requests and answers. The interactor will never output any empty lines.

Problem B. Balanced Bouldering

Input file:	*standard input*
Output file:	*standard output*
Time limit:	10 seconds
Memory limit:	512 mebibytes

After the lockdown Bruce decided to go to his favorite bouldering hall. There, he took to one of the easier walls and tried to make his way up. Unfortunately, after month of lockdown he could never quite reach the top, as he would always run out of stamina and fall down.

While climbing, he noticed the holds all have different shapes with some of them being much harder to hold than others, so gripping them uses up different amounts of stamina.

Bruce calls *balanced way* the way up that he can take without running out of stamina.

Then Bruce asks you to write a program to show him the shortest balanced way.

The bouldering wall is a rectangular grid of cells of size 1×1 where holds can be installed. For this problem we do not consider the varying sizes of the holds, so you can assume them to be the shape of a singular point exactly in the middle of the cell. Bruce can only move from one hold to another if their distance (the Euclidean distance between the centres of the cells) does not exceed his arms' reach.

Input

First line of the input contains four integers h, w, r and s ($2 \le h \le 25, 1 \le w \le 25, 1 \le r \le 3, 1 \le s \le 10^9$) where h and w are the height and width of the bouldering wall, r is the reach of Bruce's arms and s is a numerical representation of Bruce's stamina. Then h lines follow, each with w characters, describing the holds on the bouldering wall. Each character is either a digit c ($\texttt{1} \le c \le \texttt{9}$), which means that a hold with difficulty c is installed at this position, or "`.`", which means there is no hold installed.

The first line corresponds to the top of the bouldering wall and the last line to the bottom.

A sequence of holds is a valid route for Bruce if the following conditions are satisfied:

- The route starts at the bottommost hold and ends at the topmost hold. There will always be a unique bottommost and a unique topmost hold, and these are guaranteed to be distinct.
- The sum of difficulty levels of the used holds is at most s.
- The Euclidean distance between any two consecutive holds along the route is at most r.

Output

Output the total length of a shortest route Bruce can climb to reach the topmost hold without running out of stamina. Your answer should have an absolute or relative error of at most 10^{-6}. If it is not possible for Bruce to reach the top, output -1 instead.

Examples

standard input	standard output
12 12 3 11 ....3... .....3.1.. ....2.... .....2..... .1.1..... .....2..... .1..... ...2..... .1.....	13.543203766865055
8 15 3 15 .....1..... ...1..1.1..... .2.....1.... ..2.....1..... ....4.1..2..1.. ....1.....	6.414213562373095
10 9 2 10 ...2..... ....5.2... ....3... ....5.... ..2....2. ..1..... ....2.... ..1.....	-1

Problem C. Counting Creatures

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

Alice is a new director on a Space Zoo. In the zoo are **exactly** N kinds of the alien animals and Alice wants to know their quantities. She only knows that different kinds of animals have different number of legs (at least one), but she has no idea how many legs they each have. She trains the animals and tries to figure it out using the next algorithm: First she makes all the animals stand up with all their legs and counts their legs. Then, for each time she whistles, all the animals lift one leg (if it has at least one leg standing on the ground), and then she counts the feet again.

After K times, Alice thinks that it is enough to determine the quantity of each kind of animal, but does it really work? So, it is your job to help her to solve the problem.

Input

The first line contains an integer $T(1 \le T \le 100)$, indicating the number of test cases.

Each test case contains two lines. The first line contains two integers $N(1 \le N \le 1000)$ and $K(1 \le K \le 1000)$, representing the number of different kinds of animals and the time Alice whistles. The second line contains $K + 1$ integers $A_0, A_1 \dots A_K(0 \le A_i \le 10^4)$ where A_i represents the number of legs after her i^{th} whistle. You may assume that size of the input does not exceed 64 KiB.

Output

For each test case in the input, print several lines. The first line shall contain 0 if there is no solution, 1 if there is unique solution or 2 if there are multiple solutions. according to the result.

If the result is uniquely determined, you should print N extra lines each contains two integer L_i, N_i, where L_i represents how many legs does the i^{th} kind of animal have and N_i represents the number of i^{th} kind of animal. The animals should be sorted by the number of their legs in ascending order.

Examples

standard input	standard output
3 2 3 14 9 6 3 2 2 8 5 3 3 2 20 13 8	1 1 2 4 3 0 2

Problem D. Drunk Deer

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Once the drunk deer got lost in the forest.

The forest can be represented as the $N \times M$ rectangle. The southwest corner of the forest is placed at the origin, and edges are parallel to the coordinate axis (so coordinates of all points inside the forest are non-negative). The trees are placed in points with integer coordinates inside or on the border of the rectangle.

The deer starts his way near the some tree and then chooses **randomly and equiprobably** one of its four neighbors. The way between trees takes one second. If deer reaches the tree on the border, he successfully ends his journey. But there is one broken tree in the forest; if deer reaches this tree, the tree immediately falls down and kills the deer, ending his journey unsuccessfully.

Given N, M, coordinates of tree where the deer starts and coordinates of the broken tree, calculate expected time of deer's journey (regardless if it was successful or not).

Input

First line of the input contains two integers N and M ($1 \le M, N \le 700$). Second line contains two integers x_b and y_b — coordinates of the broken tree. Third line contains two integers x_d and y_d — coordinates of the starting tree for the deer ($0 \le x_b, x_d \le N$, $0 \le y_b, y_d \le M$).

Output

Print the expected time of the deer's journey in seconds with absolute or relative precision 10^{-8} or better.

Examples

standard input	standard output
2 4 0 0 1 1	1.4285714286

Problem E. Eastern Empire

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

The map of the Eastern Empire is given as an rectangle area of size $N \times M$. Each grid is an empty area, a city or an enemy. Emperor wants to enclose empire by the wall in the next way. The wall is a simple polygon build alone the edge of the grids, enclosing all the cities and keeping all the enemies out.

Emperor wants to make the wall as short as possible. Now it is your job to calculate the length of the shortest wall length so that it can protect all the cities of Empire from the enemies.

Input

The first line contains an integer $T(1 \le T \le 50)$, indicating the number of test cases.

Each test case contains several lines.

The first line contains two integer $H, W(1 \le H, W \le 8)$, indicating the number of rows and columns of the map.

The following H lines contains W chars, indicating the map. '`c`' represents a city, '`.`' represents a empty area and '`e`' represents an enemy.

You can assume that there will be at least one city on the map.

Output

For each test case in the input, print the length of the shortest wall, enclosing the Eastern Empire (-1 if impossible).

Examples

standard input	standard output
3 3 3 .c. .e. c.c 4 4 .... .ce. .ec. .... 5 5 .ccc. .e... ..ecc e.ecc .ce.e	14 -1 28

Problem F. Funny Frogs

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

Among other magical creatures, Rubeus Hagrid is breeding magical frogs. Those frogs are of a special species and only occur in one of n distinct colours. Breeding works in next way. At the evening, Hagrid puts $n-1$ frogs with distinct colours in a special breeding cage, gives them strawberry cheese and fresh carrot to eat, and lets them stay in the dark over night, then there will be y frogs in the terrarium the next morning. The crazy thing is that none of the y frogs has any of the $n-1$ original colours. Instead, all of them take on the colour that was not present in the breeding cage initially.

Last weekend Hagrid let a friend borrow his frogs for the Christopher Street Day. Of course his friend wanted the frogs to look as colourful as possible. That's why Hagrid applied his method in a way such that there were at least $n-1$ frogs available of every single colour. Therefore, Hagrid currently has $x_i \geq n-1$ frogs of the ith colour. However, the next holiday coming up is St. Patrick's Day and Hagrid thinks that it would be super cool if all his frogs took on the same colour for this special event. He is wondering whether such a state can be reached by exclusively applying the C^4 method. Because he is running low on strawberry cheese, he would like to know the minimal number of breeding nights needed for all his frogs to take on the same colour - provided that this is even possible.

Input

The input consists of:

- A line with three integers n, c, and y:
  - n ($2 \leq n \leq 10^5$), the number different colours that the frogs can take on.
  - c ($1 \leq c \leq n$), the colour that all frogs should have in the end.
  - y ($n-1 \leq y \leq 10^9$), the number of frogs in the breeding terrarium after applying the C^4 method.
- A line with n integers $x_1, \ldots, x_n$ ($n-1 \leq x_i \leq 10^9$ for all i), where x_i is the number of frogs of the ith colour that Hagrid possesses initially.

Output

If it is impossible for Hagrid's frogs to all take on colour c by exclusively applying the C^4 method, output `impossible`. Otherwise output two integer numbers a and b, where a is the minimal number of C^4 applications necessary for all frogs to have colour c, and b is the total number of frogs that Hagrid possesses in the end.

Examples

standard input	standard output
3 2 2 2 3 5	5 10
3 1 3 2 2 3	impossible

Problem G. Grid Game

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Let's introduce next grid game with balls.

Consider a rectangular board N cells in width and M cells in height. Two to four balls are placed in the cells of the board. Each ball is either black or white. At any moment of time, no two balls can occupy the same cell. Cells with no balls in them are either empty or occupied by a wall. A move consists in pushing a ball in one of the directions: up, down, left or right. After the ball is pushed, it moves in the given direction until it reaches either the edge of the board, another ball or a wall.

The goal is to make the minimal number of moves so that the balls are placed as in the given pattern. The position on the board matches the pattern if there exists a parallel shift of the pattern such that after the shift, the positions and colors of all balls in the pattern and in the labyrinth coincide. The pattern can be neither rotated nor reflected. Parts of the pattern that do not contain any balls can coincide with the exterior of the board.

Input

The input consists of a single test case. The first line of input contains two integers M and N, height and width of the board ($5 \le M, N \le 8$). The next M lines contain N characters each and describe the board. Walls are marked with a hash ('`#`'), empty cells are marked with a dot ('`.`'), black balls with digit '`0`' and white balls with digit '`1`'. The total number of balls on the board is no less than two and no more than four.

The next line contains two integers H and W, height and width of the pattern ($1 \le H, W \le 5$). The next H lines contain W characters each and describe the pattern. The colors of balls in the pattern are also marked with digits '`0`' and '`1`', all other characters of the pattern are asterisks ('`*`'). Asterisks correspond to either empty cells, walls or cells outside the board. The number and colors of balls in the pattern and on the board are equal. It is guaranteed that the initial position on the board does not match the pattern.

Output

On the first line of output, write the minimal number of moves required to solve the problem. On the next lines, describe the moves themselves. The description of a single move consists of the position (row number and column number) of a ball and the direction in which it is pushed. The rows are numbered from highest to lowest starting from 1, the columns are numbered from leftmost to rightmost starting from 1. The direction is written as follows: '`D`' means pushing down, '`U`' means pushing up, '`L`' to the left and '`R`' to the right. It is guaranteed that in all tests given to your program, the solution exists.

Examples

standard input	standard output
5 6 .....1 .####. ....#. .####. 0....0 2 2 00 1*	8 5 1 U 1 1 R 5 6 L 5 1 U 1 5 L 1 6 D 5 6 L 5 1 U
5 5 ....0 #...1 .... 3 4 **** *0** **1*	2 1 5 L 2 5 L

Problem H. Hillary's Hobby

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

Hillary has painted some rectangles with completely black surfaces on a large sheet of graph paper. To ensure that everything looks neat and tidy, all of her rectangles' edges lie on the lines of the paper. Now she intends to take a photo of the paper while spinning it very fast.

The exposure time of the camera is long enough to cover an entire rotation of the plane. In the resulting photo, areas covered by rectangles throughout the entire rotation will appear black, whereas areas never covered by a rectangle will appear white. The remaining "blurry" areas may vary in brightness, but Hillary simply considers all of those to be grey.

Now, before taking the photo, Hillary tells you the coordinates of the rectangles she has painted and asks you to determine how much of the photo's surface area will be black and how much of it will be grey.

Input

The input consists of:

- One line with an integer n ($1 \le n \le 1\,000$), giving the number of rectangles.
- n lines, each with four numbers: two integers x_{min}, y_{min} ($-10^9 \le x_{min}, y_{min} \le 10^9$) giving the coordinates of the rectangle's bottom left corner (that is, the corner with minimum x and y values) and two integers w, h ($1 \le w, h \le 200$) giving the rectangle's width and height.

The plane is rotated around the origin $(0, 0)$, and you may safely assume that the entire sheet of paper is fully captured by the camera throughout the entire rotation.

Output

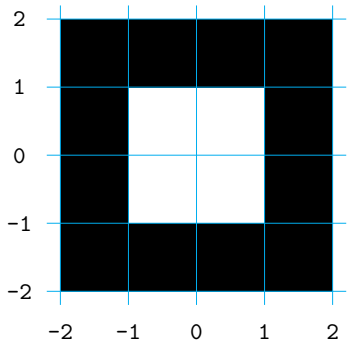
Output two real numbers, the first being the area of the black and the second being the area of the grey portion of the photo's surface. A number is considered correct if its absolute or relative error does not exceed 10^{-6}.

Examples

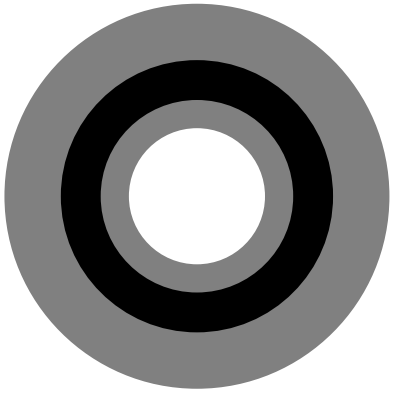
standard input	standard output
4 -2 1 4 1 -2 -2 1 4 -2 -2 4 1 1 -2 1 4	6.2831853 15.7079633
1 0 0 2 2	0.0 25.1327412

Note

The first sample:



The resulting photo:



Problem I. Interesting Integers

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

Determine whether there exists n pairwise distinct **odd** positive integers a_i such as $\sum_{i=1}^{n} \frac{1}{a_i} = 1$.

Input

Input consists of one line containing an integer N ($1 \le n \le 100$).

Output

Print n positive integers a_i, separated by spaces. Those integers should not exceed 10^{18} by absolute value. If no such set exists, print -1 instead. If there are multiple solutions, print any of them.

Examples

standard input	standard output
1	1
2	-1

Problem J. John's Jigsaw

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	512 mebibytes

John likes to solve jigsaw puzzles. So when he planned to create the rectangular wall, he ordered the stones he plans to use to create the rectangular wall and sorted them by size.

John does not bother about the exact dimensions, the wall must be rectangular without holes and use all the stones John ordered.

All of them have the same width, but the length and height may vary. For each stone however, its length is equal to its height and both are a power of two. The width of the wall must be equal to the width of the stones, i.e. you may not rotate the stones and you may not put two stones behind each other.

Given the side lengths and quantities of the stones, determine if it is possible to build a wall using all available stones, and if so, what length and height such a wall might have.

Input

The input consists of:

- One line with an integer n ($0 \le n \le 25$), indicating that the largest stone has a side length of 2^n.
- One line with $n+1$ integers $m_0, \ldots, m_n$ ($0 \le m_i \le 10^{15}$ for each i, $m_n \ge 1$), where m_i describes the number of stones of side length 2^i.

It is guaranteed that the combined area of all stones is at most 10^{15}.

Output

If it is possible to build a rectangular wall without any holes, output one line with two integers, the length and height of a possible wall that can be built. Otherwise, output `impossible`. If multiple solutions exist, output any of them.

Examples

standard input	standard output
2 21 3 3	9 9
2 0 3 2	impossible

Problem K. King's Keyword

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

The history of math in Byteland is quite remarkable. In the middle ages, every New Year the King declared the keyword of the year. Then the royal Numerologists calculated lucky number based on King's keyword and all calculations during this year had to be done in a positional numeral system with this lucky number as the base.

Naturally this lead to a lot of confusion throughout the years. On the flip side it makes determining the year of ancient documents really easy for historians.

Recently an old manuscript was discovered which only features a simple multiplication of two numbers. As there are no other clues about the year of origin, some historians have asked you for help to determine the base of the system the calculation was performed in.

Input

The input consists of three lines, each describing a positive integer with no leading zeroes. Each line consists of:

- One integer n ($1 \le n \le 1\,000$), the number of digits of the currently described integer.
- n integers $d_{n-1}, \ldots, d_0$ ($0 \le d_i \le 2^{30}$ for each i, $d_{n-1} \ne 0$), the digits of the integer. The most significant digit is d_{n-1}, the least significant digit is d_0.

The first two lines correspond to the factors, the third line to the product of the multiplication.

Output

Output a possible base the multiplication was performed in. If there are multiple possible bases, you may output any of them. If no possible base exists, output `impossible`.

Examples

standard input	standard output
2 2 0 1 2 3 1 0 0	4
3 5 1 2 2 11 3 5 4 5 1 12 6	13
2 3 2 2 3 2 3 10 12 4	impossible

Problem L. Laura's List

Input file:	*standard input*
Output file:	*standard output*
Time limit:	4 seconds
Memory limit:	512 mebibytes

Laura is working on a word list containing N words. She has so poor a memory that it is too hard for her to remember all of the words on the list. But she does find a way to help self to remember. She finds that if a sequence of words has a property that for all pairs of neighboring words, the previous one is a substring of the next one, then the sequence of words is easy to remember.

So she decides to eliminate some words from the word list first to make the list easier for her. Meantime, she doesn't want to miss the important words. She gives each word an importance, which is represented by an integer ranging from -1000 to 1000, then he wants to know which words to eliminate to maximize the sum of the importance of remaining words. Negative importance just means that Laura thought it useless and is a waste of time to recite the word.

Note that although she can eliminate any number of words from the word list, she can never change the order between words. In another word, the order of words appeared on the word list is consistent with the order in the input. In addition, a word may have different meanings, so it can appear on the list more than once, and it may have different importance in each occurrence.

Input

The first line contains an integer $T(1 \le T \le 50)$, indicating the number of test cases. Each test case contains several lines. The first line contains an integer $N(1 \le N \le 2 \times 10^4)$, indicating the number of words. Then N lines follows, each contains a string S_i and an integer W_i, representing the word and its importance. S_i contains only lowercase letters. You can assume that the total length of all words will not exceeded 3×10^5.

Output

For each test case in the input, print one integer — the largest importance of the remaining sequence of words.

Examples

standard input	standard output
1 5 a 1 ab 2 abb 3 baba 5 abbab 8	14

Problem M. Modern Methodics

Input file:	*standard input*
Output file:	*standard output*
Time limit:	10 seconds
Memory limit:	512 mebibytes

Your university plans to give a series of lectures on important historical events, one event per lecture, in some order.

The historical departament decided to use modern metodics whis someway reflects the structure of events in structure of course. Each event lasted for some time interval $[a_i, b_i]$. We say that two events are related, if their intervals have a common point. It would be convenient to schedule lectures on related events close to each other. Moreover, lectures on unrelated events should be given in the order in which the events have taken place (if an event A preceded an unrelated event B, then the lecture on A should precede the lecture on B).

Find the minimum integer $k \geq 0$ and an order of the lectures such that any two related events are scheduled at most k lectures apart from each other (lectures number i and j are considered to be $|i - j|$ lectures apart).

Input

The first line of input contains the number of test cases T. ($1 \leq T \leq 12345$). The descriptions of the test cases follow:

The first line of each test case contains the number n, $1 \leq n \leq 5 \cdot 10^4$. Each of the next n lines contains two integers a_i and b_i , $-10^9 \leq a_i \leq bi \leq 10^9$ — the ends of the i-th interval. The intervals are pairwise different.

You may assume that size of the input file does not exceed 4.5 MiB.

Output

Print the answers to the test cases in the order in which they appear in the input. The first line of the answer to each test case should contain the minimum value of k. The next n lines should list the intervals (in the same format as in the input) in an order such that any two related events are scheduled at most k lectures apart. Remember to put any unrelated intervals in the proper order!

Examples

standard input	standard output
1 3 1 6 2 3 4 5	1 2 3 1 6 4 5