Problem A. Rich

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2.5 seconds
Memory limit:	512 mebibytes

You are a rich person, and you think your wallet is too heavy and full now. So you want to give me some money by buying a lovely pusheen sticker which costs p dollars from me. To make your wallet lighter, you decide to pay exactly p dollars by as many coins and/or banknotes as possible.

For example, if $p = 17$ and you have two \$10 coins, four \$5 coins, and eight \$1 coins, you will pay it by two \$5 coins and seven \$1 coins. But this task is incredibly hard since you are too rich and the sticker is too expensive and pusheen is too lovely, please write a program to calculate the best solution.

Input

The first line contains an integer T indicating the total number of test cases. Each test case is a line with 11 integers p, c_1, c_5, c_{10}, c_{20}, c_{50}, c_{100}, c_{200}, c_{500}, c_{1000}, c_{2000}, specifying the price of the pusheen sticker, and the number of coins and banknotes in each denomination. The number c_i means how many coins/banknotes in denominations of i dollars in your wallet.

Limitations:

- $1 \le T \le 2 \cdot 10^4$
- $0 \le p \le 10^9$
- $0 \le c_i \le 10^5$

Output

For each test case, print the maximum number of coins and/or banknotes you can pay for exactly p dollars in a line. If you cannot pay for exactly p dollars, please simply output -1.

Examples

standard input	standard output

3
17 8 4 2 0 0 0 0 0 0 0
100 99 0 0 0 0 0 0 0 0 0
2015 9 8 7 6 5 4 3 2 1 0
	9
	-1
	36

Problem B. Non-Negative Integers

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

Marry likes to count the number of ways to choose two non-negative integers a and b less than m to make $a \times b \bmod m \neq 0$.

Let's denote $f(m)$ as the number of ways to choose two non-negative integers a and b less than m to make $a \times b \bmod m \neq 0$.

b a	0
0	0

Table 1: $a \times b$ mod 1

b a	0	1
0	0	0
1	0	1

Table 2: $a \times b$ mod 2

b a	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

Table 3: $a \times b$ mod 3

b a	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	3	4	5
2	0	2	4	0	2	4
3	0	3	0	3	0	3
4	0	4	2	0	4	2
5	0	5	4	3	2	1

Table 4: $a \times b$ mod 6

She has calculated a lot of $f(m)$ for different m, and now she is interested in another function $g(n) = \sum_{m|n} f(m)$. For example, $g(6) = f(1) + f(2) + f(3) + f(6) = 0 + 1 + 4 + 21 = 26$. She needs you to double check the answer.

Give you n. Your task is to find $g(n)$ modulo 2^{64}.

Input

The first line contains an integer T indicating the total number of test cases. Each test case is a line with a positive integer n.

Limitations:

- $1 \leq T \leq 2 \cdot 10^4$
- $1 \leq n \leq 10^9$

Output

For each test case, print one integer s, representing $g(n)$ modulo 2^{64}.

Examples

standard input	textslstandard output
2 6 514	26 328194

Problem C. Gameplay

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

Andy and Andrew are very smart guys and they like to play all kinds of games in their spare time. The most amazing thing is that they always find the best strategy, and that's why they feel bored again and again. They just invented a new game, as they usually did.

At the beginning of the game, they write down one string $S = s_1s_2s_3 \dots s_k$, and then they take turns(Andy first) to either:

1. Erase the leftmost character from S, that is, $S = s_2s_3s_4 \dots s_k$.
2. Erase the rightmost character from S, that is, $S = s_1s_2s_3 \dots s_{k-1}$.

Whenever S is empty or $S \in A$ (A is a given list of strings), the player which plays next loses the game.

For example, let $S = dzxx$ and $A = \{z, dz\}$. If Andy erases 'x' then Andrew can erase another 'x', because $S = dz$ and "dz" is in A, Andy, the next player, loses. Otherwise, Andy erases 'd', and then Andrew can erase 'z' result in a losing position for Andy.

You are given a string $T = t_1t_2t_3 \dots t_n$ and a list of string $A = \{a_1, a_2, \dots, a_m\}$. Your task is to find who is the winner if S is some substring of T. Andy and Andrew play so many times so you need to answer multiple queries.

Input

The first line contains an integer T indicating the total number of test cases. The following lines describe a test case.

The first line of each case contains three integers n, m, q, the length of T, the size of A, and the number of queries.The second line contains a string, representing T. Next m lines, each line consists of a string, representing a_i. Next q lines, each line consists of two integers l, r, representing a query that you should output who is the winner if $S = t_lt_{l+1} \dots t_r$.

Limitations:

- $1 \le T \le 21$
- $1 \le n, q \le 40000$
- $1 \le m \le 10000$
- $1 \le \sum_{i=1}^{m} |a_i| \le 10000$
- $1 \le l \le r \le n$
- T and strings in A consist of lowercase English letters.
- There are at most 6 test cases with $n > 5000$.

Output

For each query, if Andy wins, print 1 on a single line, otherwise print 0 on a single line.

Examples

standard input	standard output
1 10 4 10 zzzabcdzxx a z dz abcd 1 3 1 4 3 6 3 7 3 8 3 9 4 4 4 5 5 5 7 10	0 1 1 1 0 1 0 1 1 0

Problem D. Pipes Selection

Input file:	*standard input*
Output file:	*standard output*
Time limit:	8 seconds
Memory limit:	512 mebibytes

Mario works at a factory. There are n pipes in a row. Let us label the pipes $1, 2, 3, \ldots, n$ from left to right. The factory will deliver a_i units of water per second through pipe i for i from 1 to n. His job is to open some consecutive pipes to make them output exactly j units of water per second, but he doesn't know how to do it. Help Mario to find which segment of pipes to open.

You are given $a_1, a_2, a_3, \ldots, a_n$. Let $s = \sum\limits_{i=1}^{n} a_i$. Your task is to find (l_j, r_j) for all j from 1 to s such that $j = a_{l_j} + a_{l_j+1} + \ldots + a_{r_j}$. Because of his boss' command, if there are k possible (l, r) for j, then (l_j, r_j) is the $\lfloor \frac{k+1}{2} \rfloor$-th smallest one of all possible (l, r). If there are no possible (l, r) for j, then $(l_j, r_j) = (0, 0)$.

We say (x, y) is smaller than (z, w) if $x < z$ or ($x = z$ and $y < w$).

For example, $n = 4$ and (a_1, a_2, a_3, a_4)=(2, 1, 1, 2), then we can find ((l_1, r_1), (l_2, r_2), (l_3, r_3), (l_4, r_4), (l_5, r_5), (l_6, r_6))=((2, 2), (2, 3), (1, 2), (1, 3), (0, 0), (1, 4)).

There are 2 possible (l, r) for 1 which are $(2, 2), (3, 3)$ and (l_1, r_1) is the 1-th smallest, so $(l_1, r_1) = (2, 2)$.

There are 3 possible (l, r) for 2 which are $(1, 1), (2, 3), (4, 4)$ and (l_2, r_2) is the 2-th smallest, so $(l_2, r_2) = (2, 3)$.

There are 2 possible (l, r) for 3 which are $(1, 2), (3, 4)$ and (l_3, r_3) is the 1-th smallest, so $(l_3, r_3) = (1, 2)$.

There are 2 possible (l, r) for 4 which are $(1, 3), (2, 4)$ and (l_4, r_4) is the 1-th smallest, so $(l_4, r_4) = (1, 3)$.

There are no possible (l, r) for 5, so $(l_5, r_5) = (0, 0)$.

There is 1 possible (l, r) for 6 which is $(1, 4)$ and (l_6, r_6) is the 1-th smallest, so $(l_6, r_6) = (1, 4)$.

Input

The first line contains an integer t indicating the total number of test cases. The following lines describe a test case.

The first line of each case contains one integer n, the number of pipes. The second line contains n integers, representing $a_1, a_2, a_3, \ldots, a_n$.

Limitations:

- $1 \le t \le 20$
- $0 \le \min(a_i)$
- $1 \le \max(n, s) \le 30000$
- There are at most 5 test cases with $\max(n, s) > 10000$.

Output

For each test case, output on a single line two integers $\sum\limits_{j=1}^{s}((233)^j \times l_j)$ modulo 10^9+7 and $\sum\limits_{j=1}^{s}((233)^j \times r_j)$ modulo 10^9+7.

Examples

standard input	standard output

3
4
2 1 1 2
4
0 1 0 0
6
2 3 2 3 2 1
	685473415 769026629
	233 932
	811854151 883301517

Problem E. Rebuild

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

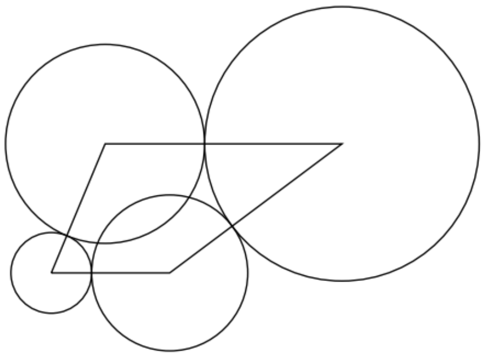
Archaeologists find ruins of Ancient ACM Civilization, and they want to rebuild it.

The ruins form a closed path on an $x - y$ plane, which has n endpoints. The endpoints locate on (x_1, y_1), (x_2, y_2), $\ldots$, (x_n, y_n) respectively. Endpoint i and endpoint $i-1$ are adjacent for $1 < i \le n$, also endpoint 1 and endpoint n are adjacent. Distances between any two adjacent endpoints are positive integers.

To rebuild, they need to build one cylindrical pillar at each endpoint, the radius of the pillar of endpoint i is r_i. All the pillars perpendicular to the x-y plane, and the corresponding endpoint is on the centerline of it. We call two pillars are adjacent if and only if two corresponding endpoints are adjacent. For any two adjacent pillars, one must be tangent externally to another, otherwise it will violate the aesthetics of Ancient ACM Civilization. If two pillars are not adjacent, then there are no constraints, even if they overlap each other.

Note that r_i must not be less than 0 since we cannot build a pillar with negative radius and pillars with zero radius are acceptable since those kind of pillars still exist in their neighbors.

You are given the coordinates of n endpoints. Your task is to find $r_1, r_2, \ldots, r_n$ which makes sum of base area of all pillars as minimum as possible.


Example

For example, if the endpoints are at $(0, 0)$, $(11, 0)$, $(27, 12)$, $(5, 12)$, we can choose (r_1, r_2, r_3, r_4)=(3.75, 7.25, 12.75, 9.25). The sum of base area equals to $3.75^2\pi$+$7.25^2\pi + 12.75^2\pi + 9.25^2\pi = 988.816\ldots$. Note that we count the area of the overlapping parts multiple times.

If there are several possible to produce the minimum sum of base area, you may output any of them.

Input

The first line contains an integer t indicating the total number of test cases. The following lines describe a test case.

The first line of each case contains one positive integer n, the size of the closed path. Next n lines, each line consists of two integers (x_i, y_i) indicate the coordinate of the i-th endpoint.

Limitations:

- $1 \le t \le 100$
- $3 \le n \le 10^4$
- $|x_i|, |y_i| \le 10^4$

Distances between any two adjacent endpoints are positive integers.

Output

If such answer doesn't exist, then print on a single line "`IMPOSSIBLE`" (without the quotes). Otherwise, in the first line print the minimum sum of base area, and then print n lines, the i-th of them should contain a number r_i, rounded to 2 digits after the decimal point.

If there are several possible ways to produce the minimum sum of base area, you may output any of them.

Example

standard input	standard output

3
4
0 0
11 0
27 12
5 12
5
0 0
7 0
7 3
3 6
0 6
5
0 0
1 0
6 12
3 16
0 12
	988.82
	3.75
	7.25
	12.75
	9.25
	157.08
	6.00
	1.00
	2.00
	3.00
	0.00
	IMPOSSIBLE

Problem F. Sorted Array

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

We are all familiar with sorting algorithms: quick sort, merge sort, heap sort, insertion sort, selection sort, bubble sort, etc. But sometimes it is an overkill to use these algorithms for an almost sorted array.

We say an array is sorted if its elements are in non-decreasing order or non-increasing order. We say an array is almost sorted if we can remove exactly one element from it, and the remaining array is sorted. Now you are given an array $a_1, a_2, \ldots, a_n$, is it almost sorted?

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with an integer n in one line, then one line with n integers $a_1, a_2, \ldots, a_n$.

Limitations:

- $1 \le T \le 2000$
- $2 \le n \le 10^5$
- $1 \le a_i \le 10^5$
- There are at most 20 test cases with $n > 1000$.

Output

For each test case, please output “YES” if it is almost sorted. Otherwise, output “NO” (both without quotes).

Example

standard input	standard output

3
3
2 1 7
3
3 2 1
5
3 1 4 1 5
	YES
	YES
	NO

Problem G. Dancing Stars on Me

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

The sky was brushed clean by the wind and the stars were cold in a black sky. What a wonderful night. You observed that, sometimes the stars can form a regular polygon in the sky if we connect them properly. You want to record these moments by your smart camera. Of course, you cannot stay awake all night for capturing. So you decide to write a program running on the smart camera to check whether the stars can form a regular polygon and capture these moments automatically.

Formally, a regular polygon is a convex polygon whose angles are all equal and all its sides have the same length. The area of a regular polygon must be nonzero. We say the stars can form a regular polygon if they are exactly the vertices of some regular polygon. To simplify the problem, we project the sky to a two-dimensional plane here, and you just need to check whether the stars can form a regular polygon in this plane.

Input

The first line contains a integer T indicating the total number of test cases. Each test case begins with an integer n, denoting the number of stars in the sky. Following n lines, each contains 2 integers x_i, y_i, describe the coordinates of n stars.

Limitations:

- $1 \le T \le 300$
- $3 \le n \le 100$
- $-10^4 \le x_i, y_i \le 10^4$
- All coordinates are pairwise distinct.

Output

For each test case, please output "`YES`" if the stars can form a regular polygon with n vertices. Otherwise, output "`NO`" (both without quotes).

Example

standard input	standard output

3
3
0 0
1 1
1 0
4
0 0
0 1
1 0
1 1
5
0 0
0 1
0 2
2 2
2 0
	NO
	YES
	NO

Problem H. Partial Tree

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

In mathematics, and more specifically in graph theory, a tree is an undirected graph in which any two nodes are connected by exactly one path. In other words, any connected graph without simple cycles is a tree.

You find a partial tree on the way home. This tree has n nodes but lacks of $n-1$ edges. You want to complete this tree by adding $n-1$ edges. There must be exactly one path between any two nodes after adding. As you know, there are n^{n-2} ways to complete this tree, and you want to make the completed tree as cool as possible. The *heatness* of a tree is the sum of *heatness* of its nodes. The heatness of a node is $f(d)$, where f is a predefined function and d is the degree of this node. What's the maximum heatness of the completed tree?

Input

The first line contains an integer T indicating the total number of test cases. Each test case starts with an integer n in one line, then one line with $n-1$ integers $f(1), f(2), \ldots, f(n-1)$.

Limitations:

- $1 \leq T \leq 2015$
- $2 \leq n \leq 2015$
- $0 \leq f(i) \leq 10000$
- There are at most 10 test cases with $n > 100$.

Output

For each test case, please output the maximum coolness of the completed tree in one line.

Example

standard input	standard output

2
3
2 1
4
5 1 4
	5
	19

Problem I. Interesting puzzle

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

Andrew is a puzzle lover, he is interested in creating interesting puzzles. One day he creates a new puzzle, following are the rules of the game:

The game is played on a big chessboard, it has n rows and m columns. The coordination (i, j) $(1 \le i \le n$ $,1 \le j \le m)$ indicates the square on i-th row and j-th column.

There are two kinds of chesses, black chess and white chess in this game. Some of the chesses are putted on some squares at the initial of the game, and other squares are empty.

The player has to put chesses on all the empty squares. Both black and white chesses are sufficient enough to be putted on all the squares.

After all the squares on chessboard are full of chess. Andrew begins to calculate the score that player gets. The score is calculated using the following algorithm: For every two squares (x_1, y_1) and (x_2, y_2), the player gets 1 point if the following 3 conditions are satisifed:

- There is a black chess on (x_1, y_1).
- And a white chess on (x_2, y_2).
- $|x_1 - x_2| = a$ and $|y_1 - y_2| = b$, which a and b are two constants defined by Andrew before the game starts.

But Andrew is confused because he doesn't know what is the highest possiple score in the game. Can you help him find the best way to solve the puzzle?

Input

The first line of input contains an integer T indicating the total number of test cases.

The first line of each test case has 4 integers n, m, a, b, indicating the number of rows, number of columns and two integers decided by Andrew. The n lines that follow describe the puzzle, the i-th of these lines is a string with m characters $c_{i,1}, c_{i,2}, ..., c_{i,m}$, which $c_{i,j}$ indicates the type of chess that putted on (i, j) at the begin of game. Black chesses are represented by uppercase 'B', white chesses by uppercase 'W', and empty squares by period '.'.

Limitations:

- $1 \le T \le 1000$
- $1 \le n, m \le 100$
- $1 \le a \le n$, $1 \le b \le m$
- There are at most 20 testcases with $n > 30$ or $m > 30$

Output

For each test case, please output an integer in the first line indicating the highest score that player can get. Then following n lines, each line with m characters indicating the chess type on each square to get the higgest score. The output format is same as input format.

Note that if there are mutiple ways to get the highest score, you should output the one with smallest lexicographical order in ASCII. ('B' is smaller than 'W').

Example

standard input	standard output
2 2 2 1 1 .. .. 2 2 1 1 .W .B	2 BB WW 2 WW BB

Problem J. John's CPU Factory

Input file:	*standard input*
Output file:	*standard output*
Time limit:	12 seconds
Memory limit:	512 mebibytes

John is a manager of a CPU chip factory, the factory produces lots of chips everyday. To manage large amounts of products, every processor has a serial number. More specifically, the factory produces n chips today, the i-th chip produced this day has a serial number s_i.

At the end of the day, he packages all the chips produced this day, and send it to wholesalers. More specially, he writes a checksum number on the package, this checksum is defined as below:

$$\max_{i,j,k}(s_i + s_j) \oplus s_k$$

which i, j, k are three different integers between 1 and n. And $\oplus$ is symbol of bitwise XOR.

Can you help John calculate the checksum number of today?

Input

The first line of input contains an integer T indicating the total number of test cases.

The first line of each test case is an integer n, indicating the number of chips produced today. The next line has n integers $s_1, s_2, .., s_n$, separated with single space, indicating serial number of each chip.

Limitations:

- $1 \leq T \leq 1000$
- $3 \leq n \leq 1000$
- $0 \leq s_i \leq 10^9$
- There are at most 10 testcases with $n > 100$

Output

For each test case, please output an integer indicating the checksum number in a line.

Example

standard input	standard output

2
3
1 2 3
3
100 200 300
	6
	400

Problem K. Maximum Spanning Forest

Input file:	`standard input`
Output file:	`standard output`
Time limit:	7 seconds
Memory limit:	512 mebibytes

In graph theory, a maximum spanning tree is a subgraph that is a tree and connects all the vertices with maximum weight sum. And a maximum spanning forest is the union of maximum spanning trees of each connected component in the graph.

A big undirected graph $G = (V, E)$ is given, which $V = \{(x, y) : 1 \le x, y \le 2,000,000,000;\ x, y \in \mathbb{Z}\}$ and $E = \{\}$ initially. You're task is to write a program that support operation x_1, y_1, x_2, y_2, c:

First, add some edges in G. You should add an edge between (a_1, b_1) and (a_2, b_2) with weight c if $x_1 \le a_1, a_2 \le x_2$, $y_1 \le b_1, b_2 \le y_2$ and $|a_1 - a_2| + |b_1 - b_2| = 1$.

Second, calculate the maximum spanning forest of G after edges added.

Input

The first line of input contains an integer T indicating the total number of test cases.

The first line of each test case has an integers n, indicating the number of operations. The n lines that follow describe the operations. Each of these lines has 5 integers x_1, y_1, x_2, y_2, c, separated by a single space.

Limitations:

- $1 \le T \le 500$
- $1 \le n \le 100$
- $1 \le x_1 \le x_2 \le 2,000,000,000$
- $1 \le y_1 \le y_2 \le 2,000,000,000$
- $1 \le c \le 1,000,000,000$
- There are at most 20 testcases with $n > 50$.

Output

For each operation, output a number in a line that indicates the weight of maximum spanning forest mod $10^9 + 7$.

Example

standard input	standard output

2
3
2 2 3 3 1
1 2 2 3 2
1 1 1 3 5
1
1 1 2000000000 2000000000 1000000000
	3
	8
	16
	999998642

Problem L. House Building

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

Have you ever played the video game Minecraft? This game has been one of the world's most popular game in recent years. The world of Minecraft is made up of lots of $1 \times 1 \times 1$ blocks in a 3D map. Blocks are the basic units of structure in Minecraft, there are many types of blocks. A block can either be a clay, dirt, water, wood, air, ... or even a building material such as brick or concrete in this game.

A typical world in Minecraft.

Nyanko-san is one of the diehard fans of the game, what he loves most is to build monumental houses in the world of the game. One day, he found a flat ground in some place. Yes, a super flat ground without any roughness, it's really a lovely place to build houses on it. Nyanko-san decided to build on a $n \times m$ big flat ground, so he drew a blueprint of his house, and found some building materials to build.

While everything seems goes smoothly, something wrong happened. Nyanko-san found out he had forgotten to prepare glass elements, which is a important element to decorate his house. Now Nyanko-san gives you his blueprint of house and asking for your help. Your job is quite easy, collecting a sufficient number of the glass unit for building his house. But first, you have to calculate how many units of glass should be collected.

There are n rows and m columns on the ground, an intersection of a row and a column is a 1×1 square,and a square is a valid place for players to put blocks on. And to simplify this problem, Nynako-san's blueprint can be represented as an integer array $c_{i,j} (1 \le i \le n, 1 \le j \le m)$. Which $c_{i,j}$ indicates the height of his house on the square of i-th row and j-th column. The number of glass unit that you need to collect is equal to the surface area of Nyanko-san's house(exclude the face adjacent to the ground).

Input

The first line contains an integer T indicating the total number of test cases. First line of each test case is a line with two integers n, m. The n lines that follow describe the array of Nyanko-san's blueprint, the i-th of these lines has m integers $c_{i,1}, c_{i,2}, ..., c_{i,m}$, separated by a single space.

Limitations

- $1 \leq T \leq 50$
- $1 \leq n, m \leq 50$
- $0 \leq c_{i,j} \leq 1000$

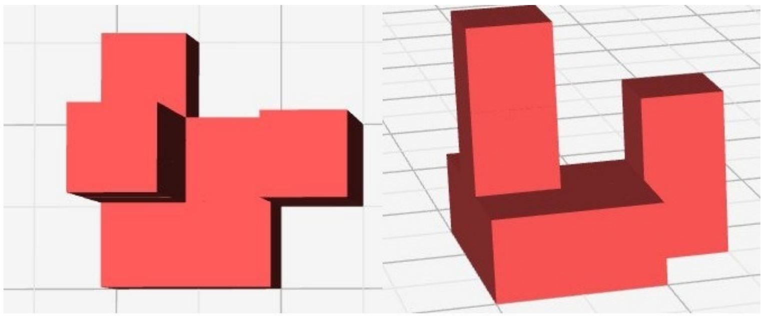
Output

For each test case, please output the number of glass units you need to collect to meet Nyanko-san's requirement in one line.

Example

standard input	standard output
2	30
3 3	20
1 0 0	
3 1 2	
1 1 0	
3 3	
1 0 1	
0 0 0	
1 0 1	

Note



A top view and side view image for sample test case 1.

Problem M. Security Corporation

Input file:	*standard input*
Output file:	*standard output*
Time limit:	40 seconds
Memory limit:	512 mebibytes

Due to the high crime rate of Bytecity, the new mayor has decided to employ security agents from c different security corporations. As a first step, he'll arrange a security agent from any of the corporation at each intersection of two roads in Bytecity.

The Bytecity has a very special and complex transportation system. There are n straight and long roads in the city, each of them can be treated as an infinitely long line on a two dimensional plane, and no 3 or more roads have a common intersection.

In order to prevent agents from slacking during work, the mayor has decided to arrange agents from two different security corporations at any two adjacent intersections. We said that two intersections are adjacent if they are on a same road, and there is no other intersection between these two intersections on that road. Also, the mayor thinks less number of security corporations is more manageable. Can you help the mayor to distribute all the intersections to as less security corporations as possible?

Input

The first line of input contains an integer T indicating the total number of test cases. The first line of each test case is an integer n, indicating the number of roads in Bytecity. The n lines that follow describes roads in Bytecity, the i-th of these lines contains 4 integers $x1_i, y1_i, x2_i, y2_i$, indicating road i is a straight line through $(x1_i, y1_i)$, $(x2_i, y2_i)$ on the plane.

Limitations:

- $1 \le T \le 1000$.
- $2 \le n \le 1000$.
- $-10^3 \le x1_i, y1_i, x2_i, y2_i \le 10^3$.
- $(x1_i, y1_i) \ne (x2_i, y2_i)$.
- There are at most 10 test cases with $n > 100$.
- There are no 2 identical roads, and at least 1 intersection exists in each test case.

Output

For each test case, please output an integer c in the first line indicating the minimum number of different corporations in your arrangement. Following $n-1$ lines, the i-th line of these lines should contain integers $a_{i,i+1}, a_{i,i+2}, ..., a_{i,n}$ in one line. The number $a_{i,j}$ indicates the security corporation that manages the intersection of road i and road j, if the intersection does not exist, $a_{i,j}$ should be -1.

Note that all the corporations should be numbered as an integer from 1 to c, and any arrangement that satisfy the mayor's requirement would be accepted.

Example

standard input	standard output

2
3
0 0 1 0
0 0 0 1
1 0 0 1
3
0 0 1 0
0 1 0 2
1 0 1 1
	3
	1 2
	3
	2
	2 1
	-1