The 2017 ACM ICPC
Asia Regional Contest
Dhaka Site

Hosted by UAP
Dhaka, Bangladesh

11th November 2017
You get 18 Pages
11 Problems &
300 Minutes

Rules for ACM ICPC Asia Regional 2017, Dhaka Site, Onsite Contest:

a) Solutions to the problems submitted for judging are called runs. Each run is judged as accepted or rejected by the judge, and the team is notified of the results.

b) Notification of the accepted runs will **NOT** be suspended at the last one hour of the contest time to keep the final result secret. Notification of the rejected runs will also continue until the end of the contest.

c) A contestant may submit a clarification request to the judges only through the CodeMarshal clarification system. If the judges agree that an ambiguity or error exists, a clarification will be issued to all contestants. Judges may prefer not to answer a clarification at all in which case that particular clarification request will be marked as IGNORED in the CodeMarshal clarification page.

d) Contestants are not to converse with anyone except members of their team and personnel designated by the organizing committee while seated at the team desk. **They cannot even talk with their team members when they are walking around the contest floor to have food or any other purpose.** Systems support staff or judges may advise contestants on system-related problems such as explaining system error messages.

e) While the contest is scheduled for a particular time length (five hours), the chief judge or the judging director has the authority to alter the length of the contest in the event of unforeseen difficulties. Should the contest duration be altered, every attempt will be made to notify contestants in a timely and uniform manner.

f) **A team may be disqualified by the** chief judge or the judging director for any activity that jeopardizes the contest such as dislodging extension cords, unauthorized modification of contest materials, distracting behavior or communicating with other teams. The **judges on the contest floor** will report to the **Judging Director** about distracting behavior of any team. **The judges can also recommend penalizing a team with additional penalty minutes for their distracting behavior.**

g) Eleven problems will be posed. So far as possible, problems will avoid dependence on detailed knowledge of a particular applications area or particular contest language. Of these problems at least one will be solvable by a first year computer science student, another one will be solvable by a second year computer science student and rest will determine the winner.

h) Contestants will have foods available in their contest room during the contest. So they cannot leave the contest room during the contest without explicit permission from the judges. **The contestants are not allowed to communicate with any contestant (even contestants of his own team) or coaches when they are outside the contest arena.**

i) Teams can bring **printed materials** with them and they can also bring five additional books. But they are not allowed to bring calculators or any machine-readable devices like CD, DVD, Pen-drive, IPOD, MP3/MP4 players, floppy disks etc. **Mobile phone MUST be switched off at all times and stored inside a bag or any other place that is publicly non visible during the entire contest time. Failure to adherence to this clause under any condition will very likely lead to strict disciplinary retaliation and possible disqualification.**

j) With the help of the volunteers, the contestants can have printouts of their codes for debugging purposes. **Passing of printed codes to other teams is strictly prohibited.**

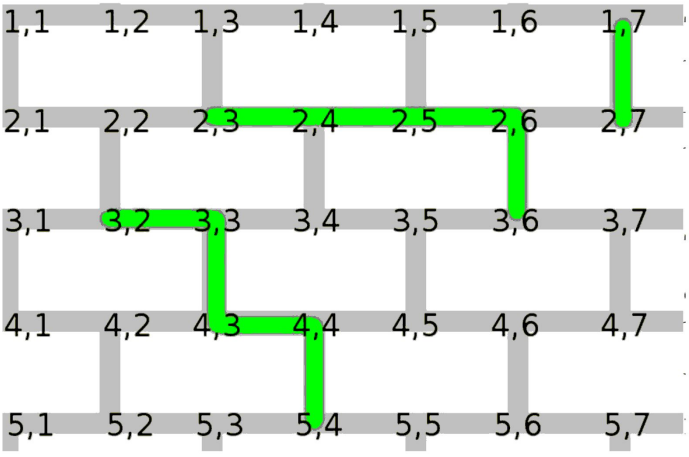
k) The decision of the judges is final.

l) Teams should inform the volunteers/judges if they don't get verdict from the codemarshal within 5 minutes of submission. Teams should also notify the volunteers if they cannot log into the CodeMarshal system. This sort of complaints will not be entertained after the contest.

Problem A

Input: Standard Input
Output: Standard Output

Brick Walls



You have to lead a pack of ants in search of food. The ants have to march through a rather large brick wall. They need to ensure that they are not on the surface of the bricks as it gets easier for mean people to crush them. If they can travel through the valleys, it would be much harder to kill them.

The bricks are laid out in a regular grid like pattern (as shown in the figure). Each brick is a rectangle that is **2** units long horizontally and **1** unit long vertically. There are tiny gaps (valleys) in between two bricks (both horizontally and vertically).

The ants are to start at a point in the valley, and their destination is a point in the valley as well. As the bricks are of fixed size and are following a regular pattern with gaps, these points can always be represented by integer coordinates.

Your task is to find the distance of the shortest path from the starting point to the destination point.

Input

There can be at most **1000** test cases. Each test case consists of four integers giving the values of starting row S_r, starting column S_c, destination row D_r, destination column D_c. You can assume that $1 \le S_r, S_c, D_r, D_c \le 10^9$. The last line of input will be "0 0 0 0" – this line must not be processed as a test case.

Output

For each test case print the distance of the shortest path in a single line.

Sample Input	Output for Sample Input
1 7 2 7 5 4 3 2 2 3 3 6 0 0 0 0	1 4 4

Problem B

Input: Standard Input
Output: Standard Output

Bracket Sequence

You are given a bracket sequence **B**. The bracket sequence may contain 4 types of brackets:

1. Round brackets (or)
2. Curly brackets { or }
3. Square brackets [or]
4. Angle brackets < or >

For each position in the bracket sequence **B**, you need to output the length of the longest balanced contiguous bracket sequence starting from (and including) that particular position.

Formally, a bracket sequence **T** is balanced if-

- **T** is empty
- **T** is **(P)**, **[P]**, **{P}**, **<P>** where **P** is a balanced bracket sequence
- **T** is **PQ** where **P** and **Q** are balanced bracket sequences.

For example, for **B = (<>)><**, the answer is "4 2 0 0 0 0".

Input

First line of the input will contain a single positive integer **T (T ≤ 10)** denoting the number of test cases. Hence **T** cases follow. Each case is a single line of non-empty bracket sequence, containing only **8** types of characters (,), {, }, [,], <, >. Each of these bracket sequences will not contain more than 10^5 characters.

If it helps, most of the judge data is produced randomly. First a random bracket sequence (not necessarily balanced) of certain length is generated. Say it is **X**. Then we change **X** by replacing some substring of it with a random balanced bracket sequence several times.

Output

For each test case, output case number (no trailing space after **Case x:**), followed by the answers in separate line. There is **NO** empty line between cases. For details, please see the sample.

Sample Input

```
5
()
<>
(<>)><
()()
{[[}
```

Output for Sample Input

```
Case 1:
2
0
Case 2:
2
0
Case 3:
4
2
0
0
0
0
Case 4:
4
0
2
0
Case 5:
0
0
0
0
```

Problem C

Input: Standard Input
Output: Standard Output

Making a Team

It is now the year **2200** and programming contest activities have spread around the world like never before. It has been estimated that there may be as many as 10^7 world class contestants in the world. Now your job is to form a team consisting of contestants only. Exactly one of the team members must be a team leader **(TL)**, exactly one must be lead developer **(LD)**, exactly one must be lead tester **(LT)**, exactly one must be marketing manager **(MM)**. Of course same person can hold more than one of these four posts. Anyone not holding any of these posts is an ordinary worker **(OW)**. Given the total number of world class contestants **N**, your task is to find out in how many ways can you form such a team. Two teams are different, if any one of the following conditions is satisfied:
i) Total number of contestants in the two teams is different.
ii) Two teams have same number of contestants and at least one contestant is different.
iii) Two teams have same contestants and at least one contestant plays different role.

For example consider the following teams:

	Member 1	Member 2	Member 3	Member 4
Team A	Contestant A (TL)	Contestant B (LD)	Contestant C (LT)	Contestant D(MM)
Team B	Contestant A (TL & MM)	Contestant B (LD)	Contestant C (LT)	Contestant D (OW)
Team C	Contestant A (TL)	Contestant B (LD)	Contestant C (LT)	Contestant E(MM)
Team D	Contestant A (TL)	Contestant B (LD)	Contestant D (MM)	Contestant C (LT)
Team E	Contestant A (TL & LD && MM)	Contestant B (OW)	Contestant D (OW)	Contestant C (LT)

Here Team A and Team B are different (although members are same) as contestant A and contestant D have difference in their roles, Team A and Team C are different because member 4 are two different persons, Team A and Team D is the same team as all members and their corresponding roles are same (only written in different order). Team E is valid because there can be more than one OW.

Input

The input file contains at most **10001** lines of input. Each line contains an integer **N (0<N<10000001)** denoting the total number of world class contestants. Input is terminated by a line containing a single zero. This line should not be processed.

Output

For each line of input produce one line of output. This line contains an integer **W** denoting the total number of ways to form a team. As this value can be too big, please output the modulo **100000007 (10^8 + 7)** value of **W** (or **W % 100000007**).

Sample Input	Output for Sample Input
2 4 100 0	18 680 95856450

Problem D

Input: Standard Input
Output: Standard Output

Christmas Tree

Christmas is coming and Bob wants to decorate his tree. The tree has **N** nodes. **1** is the root. He thinks a tree is beautiful if each node has exactly **K** child (immediate descendant) nodes (of course except the leaf nodes). He wants to remove zero or more nodes of the tree so that the property holds. If we delete a non-leaf node, the whole subtree rooted in that node, will be removed from the tree. What is the maximum number of nodes the tree can have after deleting some (possibly zero) nodes so that it has the above properties?

Input

The first line contains **T (1 ≤ T ≤ 1000)**, number of test cases. For each test case, the first line contains two space-separated integers **N** (**1 ≤ N ≤ 1000)** and **K (1 ≤ K ≤ 100)**. Each of the next **N-1** lines contains two integers **U** and **V** (**1 ≤ U, V ≤ N),** denoting an edge.

Output

For each case, print the case number and the answer.

Sample Input	Output for Sample Input
2 6 3 1 2 1 3 1 4 4 5 4 6 6 4 1 2 1 3 1 4 4 5 4 6	Case 1: 4 Case 2: 1

Problem E

Input: Standard Input
Output: Standard Output

Leap Birthdays

Do you know anyone, whose birthday is on 29th February? I don't. But I was thinking about such a person and it made me sad. A person like that would be so unlucky, don't you think? Unlike others, they will have a birthday every four years. And sometimes, not even in four years. Because we know 29th February only occurs in leap years. And a year will be a leap year if and only if the following function returns true.

```
1  bool isLeapYear(int year)
2  {
3        if(year % 400 == 0) return true;
4        else if(year % 100 == 0) return false;
5        else if(year % 4 == 0) return true;
6        else return false;
7  }
```

The above function means, a year that is divisible by 4 and not divisible by 100 are leap years but with the exception that years that are divisible by 400 are also leap years (although that is also divisible by 100).

But it's just so unfair. Take for instance a person who is born on 29th February, 1888. He will have another birthday on 29th February, 1892. Then in 1896. And then in 1900 he won't have one. Because it's a year which is not divisible by 400 but is divisible by 100. What a pity! He will have to wait 8 years for his next birthday in 1904.

So if you are given the birthday of a person and a query year **QY**, you can't easily say how many birthdays he had till 31st December of the year **QY**. There are complex mathematics involved. And it can make you very sad. So these are very dangerous problems for a human to solve. Honestly speaking, I myself got a bit sad and depressed while writing this problem thinking about those very unfortunate persons. I don't want to think of them anymore, so please solve the problem for me. Given a year **QY**, I was wondering how many birthdays a person has celebrated by the end of the year **QY**. You will also be given the person's birthday.

For example, given a person whose birthday is on 29th February, 1888, how many birthdays does he celebrate if **QY = 1910**? The answer is **4** (1892, 1896, 1904 and 1908). Similarly a person having birthday on 31stt December, 1987 celebrated **3** birthdays if **QY = 1990**. Note, a person just being born, can't celebrate his birthday. That's why we didn't count the birth year as a birthday.

Input

First line will contain an integer, **T (T ≤ 100)**, the number of test cases. Each case will contain four integers per line: **D**, **M (1 ≤ M ≤ 12)**, **Y (1850 ≤ Y ≤ 2016)** and **QY (Y ≤ QY ≤ 3000)**. Together, **D**, **M** and **Y** will form a person's birthday where **D** denotes day (**D** will always be a valid day based on **M** and **Y**), **M** denotes month and **Y** denotes year. You can assume, that it will always be a valid date. **QY** (as described above), means the year upto which you need to calculate (inclusive).

Output

Output one line per case: "Case C: X", where **C** is the case number and **X** is the answer. See the sample for clarification.

Sample Input	Output for Sample Input
4 29 2 1888 1910 29 2 1988 2010 1 1 1988 2010 31 12 1988 2010	Case 1: 4 Case 2: 5 Case 3: 22 Case 4: 22

Problem F

Input: Standard Input
Output: Standard Output

Megamind

Have you heard about Megamind? Megamind and Metro Man are two aliens who came to earth. Megamind wanted to destroy the earth, while Metro Man wanted to stop him and protect mankind. After a lot of fighting, Megamind finally threw Metro Man up into the sky. Metro Man was defeated and was never seen again.

Megamind wanted to be a super villain. He believed that the difference between a villain and a super villain is nothing but presentation. Megamind became bored, as he had nobody or nothing to fight against since Metro Man was gone. So, he wanted to create another hero against whom he would fight for recreation. But accidentally, another villain named "Hal Stewart" was created in the process, who also wanted to destroy the earth. Also, at some point Megamind had fallen in love with a pretty girl named "Roxanne Ritchi". This changed him into a new man. Now he wants to stop Hal Stewart for the sake of his love. So, the ultimate fight starts now.

- Megamind has unlimited supply of guns named "**Magic-48**". Each of these guns has **K** rounds of magic spells.
- Megamind has perfect aim. If he shoots a magic spell it will definitely hit Hal Stewart. Once hit, it decreases the energy level of Hal Stewart by **P** units.
- However, since there is exactly **K** rounds of magic spells in each of these guns, he may need to swap an old gun with a fully loaded one. This takes some time. Let's call it **swapping period**.
- Since Hal Stewart is a mutant, he has regeneration power. His energy level increases by **R** unit during a **swapping period**.
- Hal Stewart will be defeated immediately once his energy level becomes zero or negative.
- Hal Stewart initially has the energy level of **E** and Megamind has a fully loaded gun in his hand.
- Given the values of **E**, **P**, **K** and **R**, find the minimum number of times Megamind needs to shoot to defeat Hal Stewart. If it is not possible to defeat him, report that as well.

For the sake of clarity, let's take an example. Suppose, **E = 13**, **P = 4**, **K = 3** and **R = 1**. There are 3 rounds of spells in the gun. Megamind shoots all of them. Hal Stewart's energy level decreases by 12 units, and thus his energy level becomes 1. Since Megamind's gun is now empty, he will get a new gun and thus it's a **swapping period**. At this time, Hal Stewart's energy level will increase by 1 unit and will become 2. However, when Megamind shoots the next spell, Hal's energy level will drop by 4 units and will become -2, thus defeating him. So it takes 4 shoots in total to defeat Hal Stewart. However, in this same example if Hal's regeneration power was 50 instead of 1, it would have been impossible to defeat Hal.

Input

The input begins with a single positive integer **T (T ≤ 10^5)** on a line by itself indicating the number of the cases. Each of the next T lines contains four space separated integers **E**, **P**, **K** and **R** respectively, where, **1 ≤ E, P, K, R ≤ 10^5**.

Output

For each test case, output the test case number followed by the number of times Megamind needs to shoot magic spell. If it is **impossible** to defeat Hal Stewart, output **-1** instead. You should follow the exact format as the sample input/output.

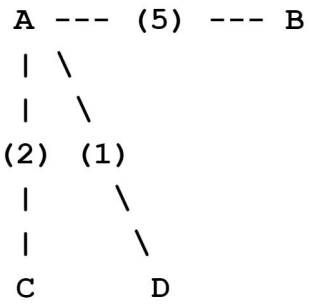
Sample Input	Output for Sample Input
4 12 4 3 2 9 4 2 7 13 4 3 1 13 4 3 50	Case 1: 3 Case 2: 4 Case 3: 4 Case 4: -1

Problem G

Input: Standard Input
Output: Standard Output

XOR Path

You are given an unrooted weighted tree. The weights on the edges are **16** bit unsigned integers, that is they are between **0** to 2^{16}**-1** (inclusive). For every integer **x** in the range **0** to 2^{16}**-1** (inclusive) find out how many pairs (unordered) of distinct nodes in the given tree have distance **x**. Distance between two nodes in the tree is defined as the bitwise xor of the edge weights on the path between these two nodes.



For example consider the tree on the right:

There are four nodes **A**, **B**, **C** and **D** in this tree. The edge weights are: **AB (5)**, **AC (2)** and **AD (1)**. So the distance between **A** and **D** is **1**, **B** and **C** is **7**, **B** and **D** is **4** etc.

Input

First line of the input contains a positive integer **T (T ≤ 10)** denoting the number of test cases. Hence **T** cases follow. Each case starts with a positive integer **n (n ≤ 100000)** denoting the number of nodes in the tree. Hence **n - 1** lines follow with the format **"u v w"** meaning there is an edge between **u** and **v (1 ≤ u, v ≤ n)** with the weight **w**.

Output

For each test case output the case number (no trailing space after Case **x:**) followed by the number of paths with the distance **x** for every **x** in the range **0** to 2^{16}**-1** (inclusive). There should **NOT** be empty line(s) between two cases. Please see the sample input output for the details.

Sample Input

```
1
4
1 2 5
1 3 2
1 4 1
```

Output for Sample Input

```
Case 1:
0
1
1
1
1
1
0
1
0
```

… (2^{16} - 9, 0s follow)

Please note, the output above is truncated intentionally to save the trees, electricity, ram consumption, network bandwidth and so on.

Problem H

Input: Standard Input
Output: Standard Output

Angry Birds Transformers

Angry Birds Transformers is a side-scrolling shoot 'em up video game, the tenth installment in the Angry Birds series, a crossover between Angry Birds and Transformers, featuring battles between the Autobirds and Deceptihogs, Angry Birds versions of the Autobots and Decepticons. It is published by Rovio Entertainment with collaboration from Hasbro. [From Wikipedia] Six consecutive screenshots from this game are shown below:

As you can see that as the player moves from left to right he can see new objects and can try to destroy them. Given the location of all the objects you will have to find out the maximum number of objects that are visible at any point of the game by the player.

For simplicity you can assume the following things:

1. All objects can be considered as points in a two dimensional Cartesian coordinate system.
2. The player runs along the x axis from left to right.
3. The viewing angle of the eye of the player is 90 degree and symmetric along the straight line $x=p_x$, where p_x is the abscissa of the player location.

The following image will make things clear:

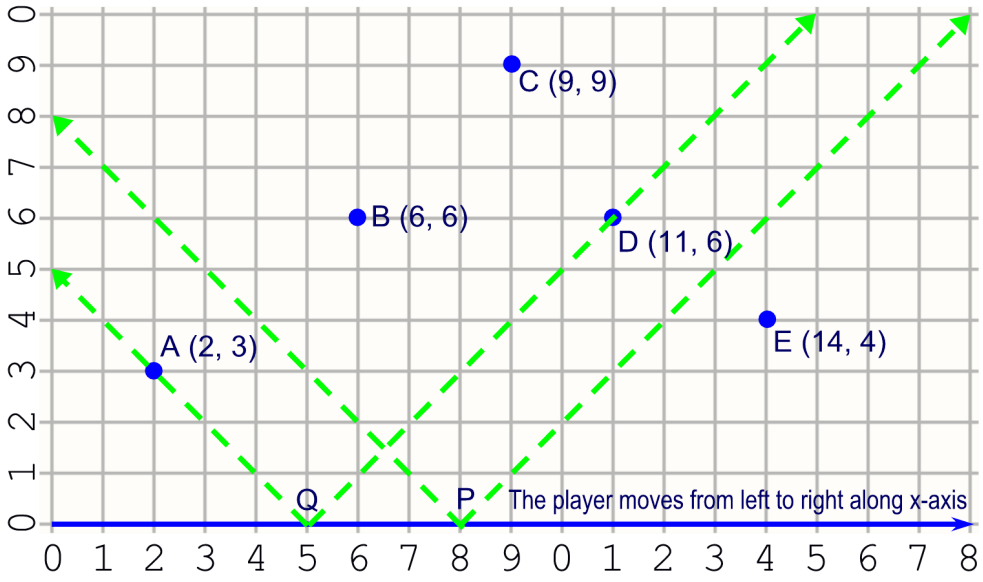


Figure: There are four objects A (2, 3), B (6, 6), C (9, 9), D (11, 6), E (14, 4). When the player is at point Q (5, 0) it can see four objects A, B, C and D and when it moves to point P (8, 0) it can see three objects B, C and D.

The goal is to find the maximum number of objects that can be seen at a time if the player moves along the x-axis. **You can assume that if two or more objects and the location of the player are collinear, the player can still see all the objects.** There is no need to find the location from where the maximum number of objects are visible as there can be more than one such place. The image above corresponds to the sample input.

Input

Input file contains at most **100** sets of inputs. Each set starts with a positive integer **N (N ≤ 10000)** denoting the total number of objects present in the scenario. Each of the next **N** lines contains two integers **(x_i, y_i)**, denoting the Cartesian coordinate of the **i^{th}** object. Here (**$0 < x_i \leq 10000$** and **$0 < y_i \leq 500$**). Input is terminated by a line containing a single zero.

Output

For each set of inputs, produce one line of output. This line contains an integer that denotes the maximum number of objects visible by a player while running along the x-axis.

Sample Input	Output for Sample Input
5 2 3 6 6 9 9 11 6 14 4 0	4

Problem I

Input: Standard Input
Output: Standard Output

Divisors

The **number of divisor** function or **d(n)** is a very interesting function in number theory. It denotes the number of positive divisors of a particular number. For example **d(24) = 8** as **24** has eight divisors **1, 2, 3, 4, 6, 8, 12** and **24**. In mathematics factorial of a positive integer number **n** is written as **n!** and is defined as below:

$$n! = 1 \times 2 \times 3 \times ... \times n = \prod_{i=1}^{n} i$$

Another interesting function **AF(n)** (Again factorial in short) is defined as:

$$AF(n) = 1! \times 2! \times 3! \times ... \times n! = \prod_{i=1}^{n} i!$$

Given **n**, your job is to find the value of **d(AF(n))**.

Input

The input file contains at most **101** lines of inputs. Each line contains an integer **n (0 < n < 5000001)**. Input is terminated by a line containing a single zero. This value should not be processed.

Output

For each line of input produce one line of output. This line contains the modulo **100000007 (10^8 + 7)** of **d(AF(n))**.

Sample Input	Output for Sample Input
1 2 3 4 100 0	1 2 6 18 59417661

Problem J

Input: Standard Input
Output: Standard Output

Substring Sorting

You are given a string, **S** (containing only lower-case letters). Next you are given some queries. The queries are of the form:

- **K M**

This means that you need to find the **M**th (1-based) substring from the list of sorted distinct substrings of **S** which has length exactly equal to **K**. For example, say **S** = "abdcabdc" and we are processing the query **K = 4**, **M = 2**, that means we are looking for substrings of length **4**. They are:

1. abdc
2. bdca
3. dcab
4. cabd
5. abdc

Since we are looking for distinct substrings, the second "abdc" will be ignored. Now if we sort them the substrings will look like:

1. abdc
2. bdca
3. cabd
4. dcab

So for **M = 2**, the output would be "bdca". However for **K = 4** and **M = 4**, the output would be "dcab". But you don't need to output the actual string. Rather just output the starting index (0-based) of the output string. If there are multiple possible answer, then output the lowest one. So for **K = 4 and M = 1** (output string "abdc"), you can see that it can be found in two different starting indices, 0 and 4. As 0 is lowest, so you need to output 0.

Input

First line will contain one integer, **T (T ≤ 10)**, number of test cases. Each case starts with a line containing **S (1 ≤ |S| ≤ 100000)**. Next line will contain **Q (1 ≤ Q ≤ 100000)**, number of queries. Each query will contain two integers **K** and **M (1 ≤ K ≤ |S|, 1 ≤ M ≤ 100000)** in a line.

Output

For each query, output the starting index (0-based) of the desired substring. If there is no answer, then output "Not found". See sample for clarification.

Sample Input	Output for Sample Input
1 abdcabdc 13	0 1 3

1 1 1 2 1 3 1 4 1 5 2 1 2 2 2 3 2 4 2 5 4 2 4 4 4 5	2 Not found 0 1 3 2 Not found 1 2 Not found

Problem K

Input: Standard Input
Output: Standard Output

Bermuda Polygon

After about five decades, people have started showing interest in the Bermuda Polygon again. Rumors say, there are some points on the earth's surface which can cause ships and aircrafts to disappear within its enclosed region under unknown circumstances. The points where the ships disappear are known as "ship-sinks", and the points causing this phenomena are "bermuda-points". Geometers and Maritime scientists from all around the globe have gathered to investigate the existence of any such thing. Little Anita, being a talented programmer and crazy about geometry is invited to be the part of this team.

Bermuda Polygon was previously known as following:
- "It is a Spherical Polygon formed by connecting some coordinates (bermuda-points) on the surface of earth such that the enclosed region contains one or more ship-sinks."

The team of scientists clarified a bit more on the matter:

- Bermuda Polygon (if exists) can only be found in the half sphere formed by all the points with latitude within range **(-90, +90)** and longitude within range **(0, +180), i.e. -90<Latitude<90 and 0<Longitude<180.**
- Every Spherical Polygon on the earth's surface can divide the earth in two bounded parts with **non zero** surface area. But Bermuda Polygon will surely be the one with the smaller surface area.

From definition, a Spherical Polygon of **K(>2)** points (each defined as $\mathbf{P_i}$) on the earth's surface is the set of following ordered geodesic line segments:

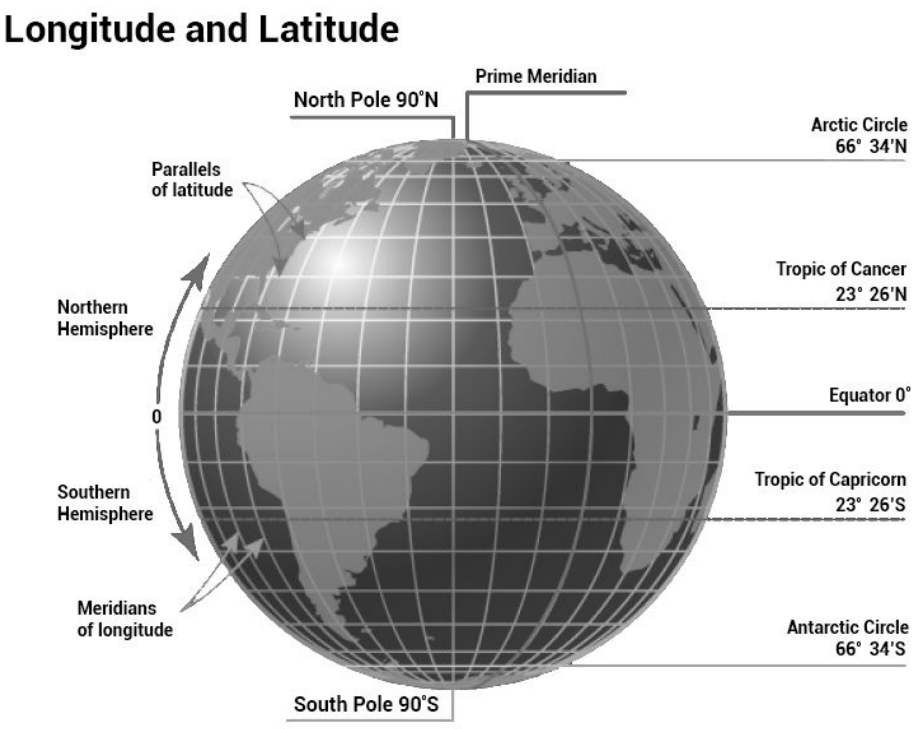
$$\{P_1P_2, P_2P_3, \dots, P_KP_1\}$$

Where, no two segments intersect with each other.
The geodesic line segment between two points on a sphere is the shortest connecting curve lying entirely on the surface of the sphere.

The team has analyzed satellite footages for last couple of years. These footages are related to ships on international water. They have found some interesting leads and pinpointed some sets of coordinates which seem to be very mysterious based on recent events. Now, they have given Anita the job to shortlist those sets where the Bermuda Polygon probably exists. As you were her "Geo" learning partner, she asks you for help.

Anita told you that, a given set of coordinates (points with latitude and longitude value) can probably indicate the existence of the Bermuda Polygon if you can draw a Spherical Polygon by taking minimum number of points from the given set such that all other points of the set are within the region enclosed by its contour (including its boundary). Here, the other points are the points of ship-sinks. Anita can't be sure about a probable Bermuda Polygon if there are no ship-sinks.

You will be given latitude and longitude of **N** points. Points will be located on one half of the earth's sphere (as clarified by the team of scientists). Radius of that sphere is $\mathbf{10^3}$**.** You have to find whether these points indicate the existence of the Bermuda Polygon.



Input

The first line will contain a single integer **D(D ≤ 150)**, denoting the number of data sets.

The first line of each data set will contain **N(3 ≤ N ≤ 200)**. Next **N** lines will contain two integers denoting Latitude and Longitude of a point respectively. Latitude is given as an angular measurement ranging from **-90°** (South Pole) to **+90°** (North Pole). Longitude is given as an angular measurement ranging from **0°** at the Prime Meridian to **+180°** eastward. The ranges are exclusive and the coordinates are unique. That means, **-90 < Latitude < 90** and **0 < Longitude < 180**.

Output

If there exists a probable Bermuda Polygon, print the **id** of the bermuda-points (the order they appear in the input, from **0** to **N-1**) in ascending order separated by spaces. If the existence of the Bermuda Polygon is inconclusive, then print "inconclusive" without quotes.

Sample Input	Output for Sample Input
2 5 40 150 30 10 -10 120 -20 20 30 30 4 0 10 0 15 0 20 0 25	0 1 2 3 inconclusive