Problem A. Perfect Cyclic String

Input:	Standard
Output:	Standard
Author(s):	Yonny Mondelo Hernández - UCI - Cuba

A perfect cyclic string is a string which can be represented as the repeated concatenation of one of its substrings. In fact, all strings can be formed in such way, and perhaps with more than one possible substring; substrings also can be the original string itself.

For example: The string "cdabcdabcdab" can be formed with the substring "cdab". The string "abababab" can be formed with the substrings "ab" or "abab", or even "abababab". And the string "qwertyuiop" can be formed only with the substring "qwertyuiop" (the string itself).

Given some string, you must find the size of the minimum substring which can be used to form the original given string.

Input

The first line contains an integer number T not greater than 10^3 representing the amount of strings to process. Each of the following T lines contains a string for processing of at most 10^3 lowercase letters of the English alphabet. You can safely assume that the sum of lengths of all given strings do not exceed $5 \cdot 10^5$.

Output

For each input string you must print an integer number in a single line representing the size of the minimum substring which can be used to form the original given string.

Example

Input	Output
5 aaaa abababab cdabcdabcdab qwertyuiopasdfg abcabcabcabcxabcabcabcabx	1 2 4 15 25

Problem B. Prop hunt!

Input: Standard
Output: Standard
Author(s): Daniel Cañizares Corrales - UCO - Colombia

Prop hunt is a game modification (or simply a mod), similar to the famous hide and seek, where two teams play against each other. One team are the props, they are players that can choose any object of the scenario to disguise themselves as, so they could be a table, a book, a barrel, a paper, whatever they want. They must be completely calm or the other team, the hunters, will kill them. As a hunter, you must be careful: if you shoot something that is not a disguised player, you start losing life. At the end, if any prop survives they will win, otherwise the hunters will win.

Your mission is to determine who is going to be the winner of the game, but thankfully, with a couple of simplifications. You will get three numbers: P - the quantity of players in the props team, H -the number of hunters and O -the objects to disguise as. Assume that all the hunters are bots (that is, controlled by the computer) and they're pretty bad: the hunters will shoot everything in the scenario, starting by the less suspicious objects and ending with the disguised players. If every hunter has 1 point of life, what team will win?

Input

The input consists of a single line that contains 3 integers P ($1 \leq P \leq 5$), H ($1 \leq H \leq 5$) and O ($1 \leq P \leq O \leq 10$).

Output

Print on a single line the text "Props win!" if the props survive, otherwise print "Hunters win!".

Example

Input	Output
2 2 3	Hunters win!
2 2 4	Props win!

Explanation: On the first case, there are 2 props and 2 hunters, but 3 objects to disguise as. The first hunter dies trying to kill the object that is not a player, but the second hunter shoots to the other objects that are the disguised players.

In the second case, the first hunter shoots a non-player object, and the second one does the same, so the props survive.

Problem C. Lucky Thief

Input:	Standard
Output:	Standard
Author(s):	Brian Giraldo Echeverri - UCO - Colombia

A very lucky thief found n keys on a street with m houses. He knows that each key opens exactly one door so he wants to know which door is, but he also wants to do the less number of tries in order to avoid the security systems.

Input

The first line contains a single integer, T ($T \leq 100000$) indicating the number of test cases. The following T lines contain two integers n and m, $1 \leq n \leq m \leq 1000000$.

Output

Print on a single line the minimum number of tries that the thief must perform in order to know what key opens each door.

Example

Input	Output
3 4 6 1 2 500 900	14 1 324750

Problem D. Making some holes

Input: Standard
Output: Standard
Author(s): Juan David Álvarez Londoño - UCO - Colombia

It's 3500 aG and our planet Earth is being represented by great athletes at the IOG (Intergalactic Olympic Games). The most exciting and hard-fought competition is the black plasmatic ball throwing. That competition takes place on an infinite flat base called RPC-314; on it, competitors throw away a really big ball made of a dense material called black plasma. All competitors are so strong that the ball will keep flying forever.

This year the black plasma was extracted from the same supernova, so it will become unstable if all the balls get close at the same position (on the ground), because they will micro-explode generating a black hole by chain reaction, which would be really catastrophic.

The game starts at some distance d of the x-axis. Each ball, say i, will always go to the right, bouncing for the first time at the coordinate $(x_i, 0)$. The i-plasma ball will bounce every $x_i - d$ units from the position where the game started. Also, because of the dragon equinox, all the balls will fly with the same x-velocity.

The Olympic Commission sent some spacecraft to catch the balls before they meet each other, but the pilots need some help: they are really neither good at physics, nor at maths, so they don't know where the balls will get together. Could you help them?

Input

The first line has an integer T ($1 \leq T \leq 10^5$), the number of test cases. The next line contains an integer n ($2 \leq n \leq 10$), the number of participants of the round, and an integer d ($1 \leq d \leq 32768$) the distance in the x-axis were the game starts. The third line has n integers, the integer x_i is the distance from the origin of the Cartesian plane where the i-plasma ball will bounce the first time on the x-axis ($d+1 \leq x_i \leq d+100$).

Output

Print on a single line the distance from the starting point where all the balls will get together.

Example

Input	Output
1 3 4 8 23 12	152

Problem E. Patty's Gift

Input: Standard
Output: Standard
Author(s): Fabio Avellaneda Pachón - Javeriana - Colombia

Last Christmas, Patty was very busy embroidering a beautiful hand-made painting for Tony. When she finished, she placed a tiny hook behind the frame so that Tony could hang it on the wall.

Tony is very bad at decorating his house, but he really likes his gift so he wants it to be placed in the best place of the house. He plans to nail two nails on the wall with a rope tied to them, and he wants to have a preview of where the painting will be placed at the end, before he ruins his wall with the holes of misplaced nails.

He has drawn a Cartesian plane on the wall, has chosen two arbitrary points (say (a, b) and (c, d)) where he pretends to nail each nail, and he plans to use a rope of an arbitrary length (say L). Now... he doesn't know where the tiny hook of his gift will be located.

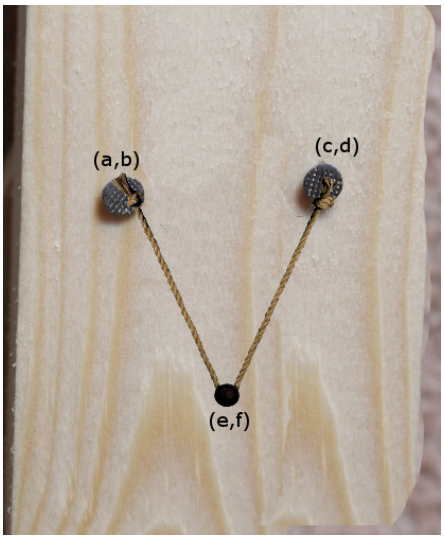
Please help Tony to visualize his new decoration, calculating where that hook will be. Assume that the painting is heavy enough to keep the rope tense, and the knots to tie it to the nails don't diminishes the length. Tony's world is very boring, so friction does not exist.

Input

The input consists of several test cases, each one in a single line. For each line you are to read the coordinates of the nails (a, b) and (c, d) and the length of the rope, L, such that $-1000 \leq a, b, c, d \leq 1000$, and $0 \leq L \leq 5000$. Each value is separated by a space.

Output

You have to calculate the coordinates (e, f) of the point where the hook will be placed, after the frame is hung on the rope. Your answer will be considered correct if the difference respect to the exact solution is less than 10^{-6}. Take into account that $-1000 \leq e, f \leq 1000$. Print the answer on a single line.



Example

Input	Output
2 5 7 4 5.099 2 5 7 4 10 10 20 30 40 31	7 4 4.788675 0.169873 11.555993 18.15728

Problem F. Triple shot

Input: Standard
Output: Standard
Author(s): Carlos Vergara Ortiz - UCO - Colombia

The Andromedan Army has invented a new kind of weapon that shoots three lasers at the same time, but with one problem: in order to impact the enemies, the ship must be at the same distance of the three objectives. You were hired to create an algorithm that calculates the point where the ship must be positioned to do a killer shot.

Input

The input consists of three lines, each one with a pair of integers X_i, Y_i ($-1000 \leq X_i, Y_i \leq 1000$), indicating the points in the plane where the enemies are placed.

Output

Print two numbers separated by a white space, indicating the point where the shoots must be performed, or "Impossible" if there's not such point. Your answer will be considered correct if the difference respect to the exact solution is less than 10^{-6}.

Example

Input	Output
-4 0 0 4 4 0	0 0
1 2 2 4 4 8	Impossible

Problem G. Crossing the river

Input: Standard
Output: Standard
Author(s): Melisa Agudelo Sánchez, Daniel Cañizares Corrales - UCO - Colombia

After a zombie apocalypse, a group of n survivors were left to die on NP-Land. For days, they fought against hunger, thirst and obviously, zombies. Pursued by the evil creatures, they traveled all over the city and finally arrived at the port. Luckily, they found a small boat, but there's a problem! The boat has only k seats, and if that wasn't enough, they will sink if the weight on board is over the limit w.

Because of that, the survivors decided to make as many trips as needed to escape from NP-Land. A trip is considered as going from the city's port to a safe island (or viceversa). There must be at least one person on each trip (in order to drive the boat).

You, as the only coder to survive the apocalypse, must help your friends to create an algorithm that minimizes the number of trips. Do it fast! The zombies are knocking on the door!

Input

The first line contains 3 integers, n ($1 \leq n \leq 16$), w ($1 \leq w \leq 100$) and k ($1 \leq k \leq 6$).

The next line will contain n integers specifying the weight of every survivor.

Output

Print on a single line an integer that represents the minimum number of travels or -1 if it's impossible.

Example

Input	Output
2 100 2 50 50	1

Problem H. The Island

Input: Standard
Output: Standard
Author(s): Brian Giraldo Echeverri - UCO - Colombia

After arriving at an island, the n survivors of the zombie apocalypse must work together to establish a base-center. There are n weapons and every survivor has more or less ability with each one. Your mission is to find a way to assign optimally the arsenal to the team.

Input

The input consists of several test cases. Each test case starts with a line that contains one integer n ($1 \leq n \leq 100$).

The following n lines will contain n integers. Line number i will specify the ability of the survivor number i using each weapon j.

Output

Print the maximum ability that can be obtained, when the weapons are optimally assigned, with an end of line.

Example

Input	Output
5 1 2 7 3 0 3 4 1 0 2 2 3 6 0 0 0 6 1 1 0 2 0 0 4 5 2 1 3 5 9 2 1 6 5 9	23 10 11

Problem I. Immortal Rabbits

Input:	`Standard`
Output:	`Standard`
Author(s):	`RPC problemsetters`

In a parallel world, a human called Fibonacci was playing with his computer. Suddenly, he got a *BSoD* (The Blue Screen of Death). While his OS restarted, he realized that a group of immortal rabbits grows as shown: $0, 1, 1, 2, 3, 5, 8, 13, 21 \ldots$, where each number is the number of pairs that were alive each month, and because of their immortality, the series could get really big in a short period of time. In other words, the number of pairs of those rabbits is the sum of the pairs of the previous two months. More formally:

$$f(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ f_{n-1} + f_{n-2} & \text{if } n \geq 1 \end{cases}$$

Trying to make a general statement, Fibonacci changed the root values (0 and 1) to variables A and B, so $f_0 = A$ and $f_1 = B$.

When the computer was finally ready, and his IDE loaded, he decided to make a program with the cool Java's BigInteger-class to calculate how many rabbits will exist in the month m. After coding the algorithm he realized it wasn't as fast as he expected, so he decided to send telepathic messages to people from other dimensions asking for a fast algorithm of big numbers to solve this problem. Our problemsetters at RPC got that message so they want to commend you that *non-trivial* task!

Input

The first line will contain an integer $T < 100$ that represents the number of test cases. The following T lines will contain 3 integers A, B, m ($0 \leq A, B \leq 1000000$, $0 \leq m \leq 100000$).

Output

Print the number of rabbits in the month m with an end of line.

Example

Input	Output
1 100 100 50	2036501107400

Problem J. Race against the clock

Input: `Standard`
Output: `Standard`
Author(s): `José Alberto Álvarez - UCO - Colombia`



We are going to simulate a competition of cycling among N participants against the clock, where each participant leaves every X units of time, and must travel a distance D. The simulation is represented by a string S with at most 300 characters. Each character represents the player's movement at each unit of time, that is, if the character C_i is 1, means that player 1 advanced at moment i ($1 \leq i \leq 300$). For easiness, only one player will advance at a time. Player number 1 leaves at moment 1, player 2 leaves at moment $1 + X$, player 3 leaves at moment $1 + 2X$, and so on. If a player that hasn't left appears, his movement must be ignored. A player finishes the race when appears at least D times after leaving.

For example, the string "122121" with $N = 2$, $X = 2$ and $D = 2$, means that player 1 advances one unit at time 1. The character number two must be ignored because the second player hasn't left, but the third character is relevant because player 2 leaves at that moment and advances one unit. The fourth character means that player one advances one more unit completing the distance D, so the time for player 1 is 4. The fifth character is 2, so the player 2 also finishes the race and his time is 3 (he finished when the total time was 5 but he left at moment 3, in other words, he was in race during characters $3, 4 and 5$ -don't forget that this race is against the clock!). The last character could be ignored.

The program should output the classification of the participants in accordance to the time registered when they get to the finish line, but there's one restriction: the simulation is valid only if all players have finished the race and travel times have been normally distributed with an index of asymmetry lesser than 20%, which is defined as follows:

$$\left| 3 \cdot \frac{(\mu - median)}{\sigma} \right|$$

Input

The first line contains an integer T, the number of test cases ($1 \leq T \leq 1000$). Each test case starts with 2 integers N ($2 \leq N \leq 6$) and D ($1 \leq D \leq 10$). The next line contains the string S with length L, representing the stopwatch. It is granted that the string only contains valid numbers, that is, $1 \leq S_i \leq N$.

After that, there is a line that contains Q ($1 \leq Q \leq L/4$). The following line will contain Q different values from X, each one separated by an space.

Output

If the arrival times of cyclists have been normally distributed, print on a single line the order of arrival,

otherwise display a message indicating that the simulation was invalid. Check the examples to see how to show that. If two or more players have the same time, order by the player's number.

Example

Input	Output
2 4 1 21243 1 1 2 2 122121111111 3 1 2 3	ORDER OF RACING: 4 1 2 3 ORDER OF RACING: 2 1 ORDER OF RACING: 2 1 INVALID SIMULATION

Problem K. Trouble in Terrorist Town

Input: Standard
Output: Standard
Author(s): Juan Felipe Cañizares Corrales - UCO - Colombia

Trouble in Terrorist Town (or simply TTT), is a known multi-player videogame where one of the players has to be a traitor, and his objective is to kill all the other players. On the other hand, the rest of the players are innocents, and they have to kill the traitor. What makes this game so interesting is the fact that nobody knows neither who the traitor nor who the innocents are. Each player knows only his/her own alignment and no one else's.

A group of friends are playing TTT in a server called: *RPC - Knuths and Barjnes* (this server is awesome). In the middle of the night the server changes the game's map. In the new map, all the players begin in a port, and they have to rent boats to go to *The Island*, a beautiful island that is full of colored pixels and nice gifts. If the innocents get to arrive to *The Island* and kill the traitor, then they win the match, and of course, all of the gifts. However, if the traitor kills all the innocent players, then he will keep the gifts and the island for himself. Besides this, there's also an extra obstacle, or difficulty, added to the traitor problem, and that is the fact that this server uses real money to rent the boats (this is a policy of the server's administrator to win money). Obviously, the players don't have tons of money to spend on the game, there for they will want to win all the prices without losing too much money.

Before the game starts, all the friends talk about how to win this match, in other words, they make a strategy. Blackcore says his strategy: minimize the cost by renting only one boat, but Marvin says to him that this is not a good strategy, because if all players go inside the same boat, then the traitor would active a C4 explosive and they all could die and lose the match. Ph.D. Sith, the most experienced player, offers a better strategy. It is common for a player not to trust in one, or even a few of his friends, because there are always some of them who are quite annoying. Those are the relationships between players. Initially all the players trust each other, until a player x says explicitly that he does not trust another particular player y.

- The players are going to make groups.
- Each group will rent only one boat.
- A group is made of players that trust each other.
- Two groups could be fused, if and only if, at least one player of the first group trusts in other player of the second group, and vice versa.
- A member of a group can invite in to other player he trusts.
- An invited player is a player that has an invitation of a group but he hasn't accepted it.
- An invited player will accept the offer, if and only if, he trusts in at least one member of the group.

- An invited player has the same rights that a member of a group, that is, he could invite in to other players that he trusts.
- If an invited player x accepts the offers, and the player y that invited him was an invited too, then y could now accept the invitation.
- A player can be part of only one group.

Ph.D. Sith has a nice idea, isn't it? But the guys have a big problem, they only get a few seconds before the match starts, and the groups have to be created just before it all begins. Your mission, if you choose to accept it, is to create a computer program that solves this problem.

Input

A test case starts with a line that contains 2 integers N ($2 \leq N \leq 5000$) and T ($0 \leq T \leq N^2 - N$), the number of players and the number of relationships, respectively.

The following T lines will contain an integer x and y that will specify the relationship between these two players, meaning that x doesn't trust in y.

The last line contains only one integer D, the cost to rent a single boat.

Output

Print the total cost of renting all boats with an end of line.

Example

Input	Output
4 6 1 4 2 1 2 3 3 4 4 1 4 2 3	3

Problem L. Points Cover

Input: Standard
Output: Standard
Author(s): Yonny Mondelo Hernández - UCI - Cuba

The Research Project for Competitions (RPC) rises again with a new mathematical task. This task is related with simple geometrical concepts, specifically with points and lines.

A point is a precise location or place on a plane, usually represented by a dot. It is important to understand that a point is not a thing, but a place. For instance we indicate the position of a point by placing a dot with a pencil. This dot may have a diameter of, say, 0.2mm, but a point has no size. No matter how far you zoomed in, it would still have no width. Since a point is a place, not a thing, it has no dimensions. If a set of points all lie in a straight line, they are called 'collinear'. If a set of points all lie on the same plane, they are called 'coplanar'. For this problem we will use coplanar points in 2D plane; for instance, points will be located using coordinates X and Y.

Then a line is a geometrical object that is straight, infinitely long and infinitely thin. It goes off in both directions forever, and is perfectly straight. A line, strictly speaking, has no ends. It has zero width. For instance, if you draw a line with a pencil, examination with a microscope would show that the pencil mark has a measurable width. The pencil line is just a way to illustrate the idea on paper. In geometry however, a line has no width. If a set of points are lined up in such a way that a line can be drawn through all of them, the points are said to be collinear.

For this problem there will be a cloud of points in the 2D plane and you must find the minimum amount of lines that must be drawn in order to cover all the points. Also you must take into account an important restriction: the lines can only be drawn parallel to the coordinate axis.

Input

An integer number $T \leq 150$ representing the number of cases and for each one: first a line with one integer $1 \leq N \leq 500$, the number of points. Then follow N lines with the coordinates of the points, not necessarily distinct. For each coordinate, two space-separated integer numbers will be given, the X and Y coordinate value respectively ($-10^9 \leq X, Y \leq 10^9$).

Output

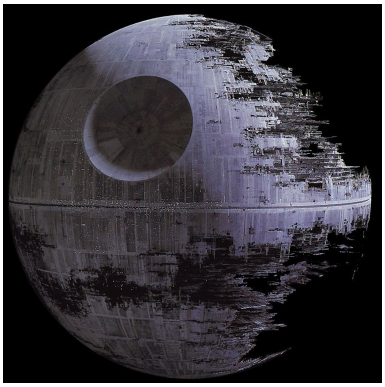
For each case you must print a line with an integer number: the minimum amount of lines parallel to coordinate axis that must be drawn in order to cover all the given points.

Example

Input	Output
3 1 3 4 4 1 1 2 1 1 2 2 2 10 1 1 1 2 2 1 2 2 2 3 2 4 3 2 4 2 3 3 4 4	1 2 4

Problem M. Death Star

Input: Standard
Output: Standard
Author(s): Lukas Restrepo Suárez, Juan Felipe Cañizares Corrales - UCO - Colombia



The *Death Star* is the greatest space station made by *The Empire*, the forces of darkness. This incredible space station could destroy one planet with just one shot of its high frequency laser and its colossal ion cannon.

The Empire is executing the order 3.1415, his target is to destroy *The moon Endor*, a beautiful natural satellite where *The Rebels* have their assault base. The *Death Star* is in position to fire, but its weapon takes around 5 minutes to reload.

The Empire has made this weapon with the shape of a five-edged star. When the weapon is loaded the five outer points of the star looks incandescent, and then they can fire and destroy their target. However, this weapon has five weak points: the difference between the star of the weapon minus the circumscribed pentagon of the middle of the same star, in other words, the triangles of the star, including the lines that form it.

You are a rebel commander, and your mission is to fly an *x-wing* and destroy the weapon of the *Death Star* shooting at the weak points.

Input

A test case starts with five lines, each one has a pairs of integers x,y ($-500 \leq x, y \leq 500$) that represents the Cartesian coordinates of the pentagon in which the weapon that has the shape of a star is inscribed. These points are in clockwise order.

The next line has an integer n ($1 \leq n \leq 100$), that represents the number of bullets fired at the weapon.

The next n lines have a pair of integer x, y that represents the Cartesian coordinate of the impact of the bullets.

Output

If the five points couldn't inscribe a star, print *Impossible*, otherwise, if a rebel's bullet impacts in one of the weak point of the weapon print *Yes*, but if the bullet doesn't impact print *No*. The answer must be printed on a single line.

Example

Input	Output
0 0 0 10 5 15 10 10 10 0 3 2 3 5 8 8 2	Yes No Yes

Credits

Additional to the problemsetters, all these people also helped with the problemset creation:

LaTeX edition

- Hugo Humberto Morales Peña
- Daniel Cañizares

Writing revision

- Ricardo Arenas
- Santiago Gutiérrez
- Sebastián Escobar

Special thanks to

- Fabio Avellaneda
- Sebastián Gómez
- Yonny Mondelo Hernández
- Juan Felipe Cañizares
- Brian Giraldo

For their constant support in testing, debugging and improving the statements, solutions and problem packages.

Problem I, Immortal Rabbits, was written by Brian Giraldo, Carlos Vergara, Daniel Cañizares (UCO) and solved by Sebastián Gómez from UTP.

Important

Writing supervisors are not responsible neither of last-minute errors that problem setters could have introduced on this document, nor the problems, nor the stories.

Pictures that appear in this problemset are property of their owners. References to Garry's Mod and Star Wars were made with educational purposes only. No copyright infringement was intended.