Problem A. Absurd Prices

Input file: `absurd.in`
Output file: `absurd.out`

Surely you know that supermarkets, shopping centres, and indeed all kind of vendors seem to have fallen in love with the digit 9, for that digit occurs most often in the price of a product, preferably at the least significant positions. Your favourite chocolate bar might cost 99 cents, just right to be able to advertise that it costs less than 1 euro. Your new bicycle might cost 499.98 euros, which, of course, is less than 500 euros.

While such comparisons are mathematically sound, they seem to impose a certain amount of stupidity on the customer. Moreover, who wants to carry home those annoying small coins you get back as change?

Fortunately, the FIFA has not adopted this weird pricing scheme: a ticket for the final in the first category for example costs 900 dollars, in the second category 600 dollars and in the third category 400 dollars. These prices may only be regarded weird for other reasons.

We want to distinguish between absurd prices like 99 cents, 499.98 euros, etc. and normal prices. To measure the absurdity of a positive integer, do the following:

- Eliminate all trailing zeros, i.e., those in the least significant positions, from the number. You now have a positive integer, say x, with a non-zero digit d at its end.
- Count the number of digits, say a, of the number x.
- If $d = 5$ the absurdity of the number is $2 \cdot a - 1$.
- Otherwise, the absurdity of the number is $2 \cdot a$.

For example, the absurdity of 350 is 3 and the absurdity of 900900 is 8. Using the measure of absurdity, we can define what we call an absurd price: A price c is absurd if and only if the closed interval $[0.95 \cdot c, 1.05 \cdot c]$ contains an integer e such that the absurdity of e is less than the absurdity of c.

Given a price in cents, go ahead and tell whether it is absurd!

Input

The first line of the input consists of the number t of test cases to follow. Each test case is specified by one line containing an integer c. You may assume that $1 \leq c \leq 10^9$.

Output

For each test case output if c is "`absurd`" or not. Adhere to the format shown in the sample output.

Sample input and output

absurd.in	absurd.out
4 99 49998 90000 970000000	absurd absurd not absurd absurd

Problem B. Cheating or Not

Input file: `cheating.in`
Output file: `cheating.out`

For the organizers of a soccer world championship the final draw is a very delicate job. It determines the compositions of the groups for the first stage of the tournament and indirectly also the possible matches in the knockout stage. The importance lies in the fact that the success of a team might depend on the opponents it faces – and, maybe, even the winner of the tournament.

The final draw is often subject to accusations of fraud. Some teams tend to think that their group is stronger than others and therefore complain they were cheated. Your job is to provide some facts that can help convince them of the opposite.

The draw is somewhat complicated due to a number of fairness considerations. The objective is not to assign too many good teams to the same group. Also teams from different confederations should be drawn into different groups. This is ensured by the following rules.

- There are g groups with m members each.
- The hosting nation will be seeded first in the first group.
- $g - 1$ selected teams will be seeded first in the remaining groups.
- The remaining positions are drawn from $m - 1$ pots, one team from each pot per group.
- You will be told which teams belong to the same confederation and you have to ensure that no two teams of the same confederation are in the same group. For confederations with more than g teams this is impossible, so for these confederations you can ignore this rule.
- You may assume that for confederations with $\leq g$ teams, all teams of the confederation which are not seeded are in the same pot.
- Note that each team belongs to exactly one confederation and each team is either seeded or contained in exactly one pot.

We want to compute the average strength of the opponents of a given team. The strengths of the teams will be given in the input. Now you have to compute the average of the sum of the strengths of the other teams in the group of the given team. The average is evaluated over all correct draws which are assumed to have the same likelihood.

Input

The input starts with the number of test cases. Each test case is described as follows.

The first line contains the number of groups $g \leq 8$ and the number of teams per group $m \leq 4$. A line with $g \times m$ integers follows. The i-th integer $0 \leq s_i \leq 10\,000$ denotes the strength of the i-th team.

The team indices start from 0. By convention, the hosting nation is assigned number 0. The next line lists the $g - 1$ seeded teams by their numbers. Each of the $m - 1$ following lines contains g teams which are allocated to the same pot.

The next line specifies the number of confederations c. Then c lines follow which describe one confederation each. Each confederation description starts with the number of teams $n_i > 0$. Then n_i numbers with the team indices follow.

The last line contains the number t of the team, whose average group strength has to be evaluated.

Output

Output the average of the sum of strengths of the opponents of team t in the group stage with three decimals on a single line.

Sample input and output

cheating.in	cheating.out
2 2 3 1 2 3 4 5 6 1 2 5 3 4 1 6 0 1 2 3 4 5 5 2 3 1 2 3 4 5 6 1 2 5 3 4 2 2 0 5 4 1 2 3 4 5	6.000 6.500

Problem C. Counterattack

Input file: `counterattack.in`
Output file: `counterattack.out`

At our soccer training camp, we have rehearsed a lot of motion sequences. In case we are defending, all players except the two strikers of our team are in our half. As soon as we are getting the ball, we are starting a counterattack with a long-range pass to one of our strikers. They know each other's motion sequences and may pass the ball to the other striker at fixed points.

There are a lot of decisions: the defender has to select the striker to pass the ball to, and the ball possessing striker has to decide at each of the n fixed points if to pass to the other striker or to run and to dribble. At the last position in the motion sequence of a striker he shoots on the goal. Each of the four actions (long-range pass, dribble, pass, and shoot on the goal) may fail (e.g. because of a defending player of the opposite team) – so our coach has assigned difficulties.

What's the minimal difficulty of a goal assuming your team plays optimally?

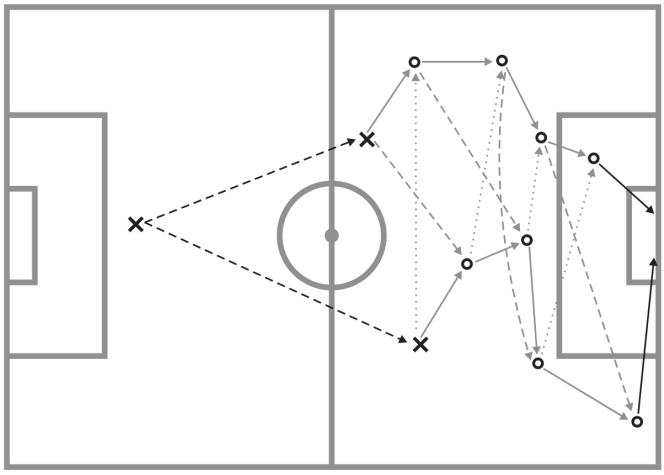


Figure 1: The defending player (cross in left half) passes the ball to one of the strikers (crosses in right half). The strikers move along fixed paths simultaneously. At each of the fixed positions (circles), the ball possessing striker either dribbles with the ball or passes to the other striker. At the last position, he shoots on the goal.

Input

The first line of the input consists of the number of test cases c that follow ($1 \le c \le 100$). Each test case consists of five lines. The first line of each test case contains n ($2 \le n \le 100\,000$), the number of fixed points in each strikers motion sequence. It is followed by l_0, l_1, s_0 and s_1, the difficulty of a long-range pass to the corresponding striker and the difficulties of the shoots of the strikers. Each striker is described in two lines (first striker 0, then striker 1): The first line contains $n-1$ difficulties, where the i-th number stands for passing from point i to the other player at point $i+1$. The second line also contains $n-1$ difficulties, where the i-th number stands for dribbling from point i to point $i+1$. You may safely assume that each difficulty is a non-negative integer less than 1 000.

Output

For each test case in the input, print one line containing the minimal difficulty of a move sequence leading to a goal.

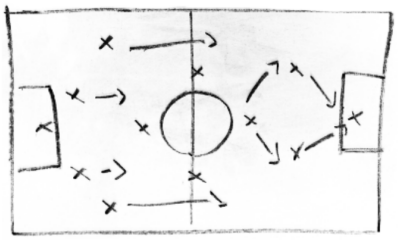
Sample input and output

counterattack.in	counterattack.out
2 3 3 5 7 999 9 13 60 5 22 6 5 5 5 3 5 7 999 9 13 8 4 60 5 17 13 22 6 15 11 5 5 18 29	23 42

Problem D. Field Plan

Input file: `field.in`
Output file: `field.out`

World Soccer Championship is coming soon and coach Yogi wants to prepare his team as well as possible. So he made up a strategy field plan for every player of the team. One plan describes a number of possible locations for the player on the field. Moreover, if Yogi wants the player to be able to move from one location A to another location B then the plan specifies the ordered pair (A, B). He is sure that his team will win if the players run over the field from one location to another using only moves of the plan.



Yogi tells every player to follow his plan and to start from a location that reaches every other location on the plan (by possibly multiple moves). However, it is quite difficult for some soccer players, simple minded as they are, to find a suitable starting location. Can you help every player to figure out the set of possible start locations?

Input

The first line gives the number of field plans. The input contains at most eleven field plans (what else?). Every plan starts with a line of two integers N and M, with $1 \le N \le 100\,000$ and $1 \le M \le 100\,000$, giving the number of locations and the number of moves. In the following M lines a plan specifies moves (A, B) by two white space separated integers $0 \le A, B < N$. The plans are separated by a blank line.

Output

For every plan print out all possible starting locations, sorted increasingly and one per line. If there are no possible locations to start, print "`Confused`". Print a blank line after each plan output.

Sample input and output

field.in	field.out
2 4 4 0 1 1 2 2 0 2 3 4 4 0 3 1 0 2 0 2 3	0 1 2 Confused

Problem E. Hacking

Input file: hacking.in
Output file: hacking.out

A coach of one of the soccer world finals teams (lets call him Hugo Hacker) wants to find out secret information about an opposing team before the game. The coach of the opposing team has a website with public information about his team. Hugo suspects that also secret information is stored on the computer which hosts the website.

The website contains a form which allows to search for key words and returns a chunk of a text file which contains the key word. Hugo has found out that by entering words which cannot be found in the documents publicly available, he can exploit a bug in the search and get access to other files on the computer. He already knows the publicly available documents. However the search box has a restriction on the maximum length of a word and the characters which can be entered. Can you tell him a word which can be entered in the search box and which does not occur as a substring in the documents?

Input

The first line of the input consists of the number of test cases which are to follow. Each test case consists of two lines: in the first line there are three integers n ($1 \le n \le 10\,000$), m ($1 \le m \le 100$) and k ($1 \le k \le 26$), where n is the length of the publicly available documents, m is the maximum allowed length of words which can be entered in the search box, and k specifies that the search box allows only the first k characters of the alphabet. The second line of each test case describes the publicly available documents and consists of n lower-case letters.

Output

For each test case in the input, print one line in the output containing a word which does not occur as a substring in the given text. The word should have at most m lower-case characters from the first k letters in the alphabet. You may assume that for each given test case, there is always at least one such word (you may print any such word).

Sample input and output

hacking.in	hacking.out
2 9 3 2 bbbaababb 9 3 2 aaabbabaa	aaa bbb

Problem F. Last Minute Constructions

Input file: `last.in`
Output file: `last.out`

For the upcoming soccer world championship's finals in South Africa the organisation committee has planned a very prestigious project. To take the two teams, which are battling it out for the title, to new heights, the final should take place on a plateau of the "Mafadi", the highest mountain of South Africa. During the preparations, the logistics of such a huge event have been severely underestimated.

Now, with barely a month to go, the stadium on top of the plateau is finished but the means of transportation to the plateau are next to nonexistent. Until now, there are only small roads connecting many little villages spread all over the mountain. Furthermore, known for their efficiency, ancient South African builders only built a road between two villages, if no other connection existed so far.

Since the amount of fans would exceed the capacity of the small mountain roads, this leaves the committee with only one choice: improve the possibilities to reach the mountain at one of the sites. But as if this wasn't enough trouble to go through, the mountain folks have announced to sabotage the finals, if the constructions would disturb any village more than once. Since the committee has access to an old tunnel-drill, it has decided to create a number of alternative routes to divert a bit of the traffic.

The engineers have identified a number of possible sites, all offering a good landing spot to fly in the giant drill to and a takeoff spot to transport the drill back from. But as the drill is really old, it has to follow the natural structures in the rock and can therefore only be used to drill in the given direction. Thus, the engineers seek your help to identify the sites on which a route for the drill (using existing roads and drilling new tunnels) exists from the landing platform to the takeoff spot, visiting each village at most once. Furthermore, a valid route needs to contain all the tunnels identified necessary by the engineers, and it should contain no other tunnels.

Input

The input to your program provided by the South African building committee will be structured as follows. Each input file begins with the number of test cases on a single line. On the first line of every test case three numbers N, M, T ($1 \le N, M \le 100\,000, 0 \le T \le 100\,000$) will specify the number of villages, as well as connections and tunnels to follow. The second line specifies the location of the landing platform and the takeoff spot respectively (landing platform $\neq$ takeoff spot). After this M lines follow, each giving a pair of villages a b ($0 \le a, b < N, a \neq b$) to indicate an existing road between a and b which can be used in both directions. Finally T lines follow, each giving a pair of villages a b ($0 \le a, b < N, a \neq b$) to indicate that a tunnel was deemed necessary for the finals from a to b. The tunnel has to be drilled in the direction from a to b.

Output

For each of the presented test cases, print a single line containing either "`IMPOSSIBLE`" whenever the construction is not possible, or "`POSSIBLE`" whenever the constructions can be carried out under the given restrictions.

Sample input and output

last.in	last.out
2 3 2 1 1 0 0 1 0 2 1 2 3 2 1 1 0 0 1 0 2 2 1	POSSIBLE IMPOSSIBLE

Problem G. Lineup

Input file: `lineup.in`
Output file: `lineup.out`

On June 13th team Germany has its first match in the FIFA world cup against team Australia. As the coach of team Germany, it is your duty to select the lineup for the game. Given this is your first game in the cup, naturally you want to make a good impression. Therefore you'd like to play with the strongest lineup possible.

You have already decided on the tactical formation you wish to use, so now you need to select the players who should fill each of the 11 positions in the team. Your assistant has selected the 11 strongest players from your squad, but this still leaves the question where to put which player. Most players have a favoured position on the field where they are strongest, but some players are proficient in different positions. Your assistant has rated the playing strength of each of your 11 players in each of the 11 available positions in your formation, where a score of 100 means that this is an ideal position for the player and a score of 0 means that the player is not suitable for that position at all. Find the lineup which maximises the sum of the playing strengths of your players for the positions you assigned them. All positions must be occupied, however, do not put players in positions they are not proficient with (i.e. have a score of 0).

Input

The input consists of several test cases. The first line of input contains the number C of test cases. For each case you are given 11 lines, one for each player, where the i-th line contains 11 integer numbers s_{ij} between 0 and 100. s_{ij} describes the i-th players strength on the j-th position. No player will be proficient in more than five different positions.

Output

For each test case output the maximum of the sum of player strengths over all possible lineups. Each test case result should go on a separate line. There will always be at least one valid lineup.

Sample input and output

lineup.in	lineup.out
1 100 0 0 0 0 0 0 0 0 0 0 0 80 70 70 60 0 0 0 0 0 0 0 40 90 90 40 0 0 0 0 0 0 0 40 85 85 33 0 0 0 0 0 0 0 70 60 60 85 0 0 0 0 0 0 0 0 0 0 0 95 70 60 60 0 0 0 45 0 0 0 80 90 50 70 0 0 0 0 0 0 0 40 90 90 40 70 0 0 0 0 0 0 0 50 70 85 50 0 0 0 0 0 0 0 66 60 0 80 80 0 0 0 0 0 0 50 50 0 90 88	970

Problem H. Polynomial Estimates

Input file: `polynomial.in`
Output file: `polynomial.out`

The number of spectators at the FIFA World Cup increases year after year. As you sell the advertisement slots during the games for the coming years, you need to come up with the price a company has to pay in order to get an advertisement slot. For this, you need a good estimate for the number of spectators in the coming games, based on the number of spectators in the past games. Your intuition tells you that maybe the number of spectators could be modeled precisely by a polynomial of degree at most 3. The task is to check if this intuition is true.

Input

The input starts with a positive integer N, the number of test cases. Each test case consists of one line. The line starts with an integer $1 \leq n \leq 500$, followed by n integers $x_1, \ldots, x_n$ with $0 \leq x_i \leq 50\,000\,000$ for all i, the number of spectators in past games.

Output

For each test case, print "`YES`" if there is a polynomial p (with real coefficients) of degree at most 3 such that $p(i) = x_i$ for all i. Otherwise, print "`NO`".

Sample input and output

polynomial.in	polynomial.out
3 1 3 5 0 1 2 3 4 5 0 1 2 4 5	YES YES NO

Problem I. Soccer Bets

Input file: bets.in
Output file: bets.out

The teams have finished the group stage of the FIFA World Cup and the teams that are in the round of the last sixteen are known. My boss has all of the games analyzed and bet on the whole rest of the tournament – writing the outcome of each match on a single sheet of paper. It was my job to bring his bets to the next betting office and set 1 000$. Being nervous with so much cash in my pockets I fell over (I am a bit clumsy) and the bets got shuffled. So I dont know if a bet corresponds to the final match or the semi-final or something else.

I do not want to disappoint my boss, so I decided to place only one bet on the winner of the tournament. Everything I know is that in each round the teams that win (a team wins if it shoots more goals than the opposing team) are in the next round, the other teams are eliminated from the tournament. This is not true for the semi-finals where the losers also play for the third place. So we have in total 16 matches.

Can you please tell me which team will win the World Cup based on the bets of my boss?

Input

The first line of the input is the number of test cases c ($1 \le c \le 100$). Each test case consists of 16 lines describing the matches in random order. A match description looks as follows: t_1 t_2 g_1 g_2. t_1 and t_2 are the names of teams (abbreviated as exactly three uppercase letters), g_1 and g_2 ($0 \le g_1, g_2 \le 10; g_1 \ne g_2$) are the goals of the two teams.

Output

For each test case, print one line containing the team that will win the FIFA World Cup (based on the analysis of my boss which is always correct!).

Sample input and output

bets.in	bets.out
1 ITA URU 2 0 ITA IRE 1 0 ITA ARG 3 4 YUG ARG 2 3 GER CZE 1 0 ENG GER 3 4 ITA ENG 2 1 CAM COL 2 1 ENG CAM 3 2 ENG BEL 1 0 GER ARG 1 0 CZE CRC 4 1 NET GER 1 2 BRZ ARG 0 1 SPA YUG 1 2 ROM IRE 4 5	GER

Problem J. The Two-Ball Game

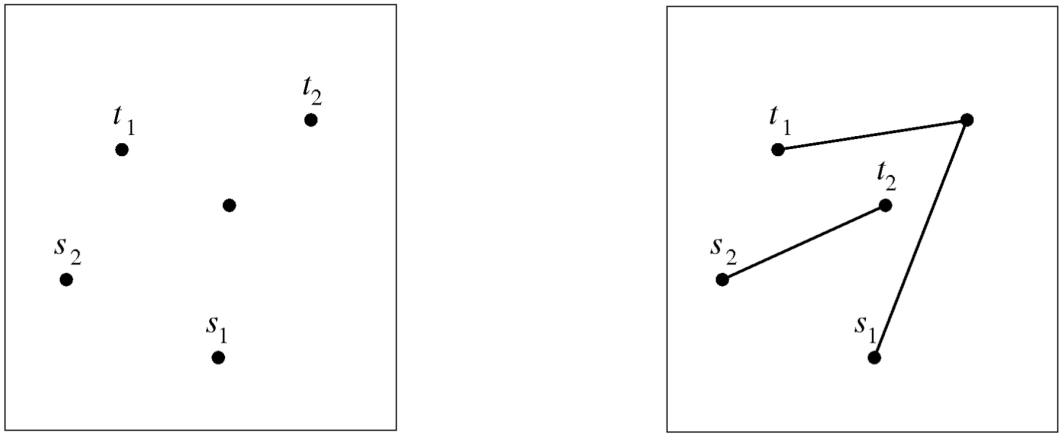
Input file: `two.in`
Output file: `two.out`

Lizarb's national soccer team undoubtedly belongs to the group of favourites to win the World Cup at the upcoming championship. Their greatest advantages are their excellent dribbling skills and the ball passing precision. Particularly, each player can pass the ball to every other player on the playing field at any distance. The team's captain, Oicul, claims that an exercise which certainly has a substantial effect on the team's soccer skills is the so-called "Two-ball Game".

In the two-ball game, $n \geq 4$ kickers are positioned on the playing field and do not move (i.e. change their locations) during the game. Four of the players are distinguished: two of them, denoted as s_1 and s_2, are called starting players, and two others, denoted as t_1 and t_2 , are called terminal players.

At the beginning, player s_1 has got a white ball and s_2 possesses a black ball. Then each starting player can kick the ball directly to the corresponding terminal player but he can also kick the ball to any other player on the field and this player can pass the ball to the next one, and so on. The aim is that at the end the white ball is in possession of t_1 and the black ball in possession of t_2. So, it seems the game is quite simple. However, to avoid ball collisions, the constraint of the game is that no ball trajectories cross each other and that no player (including starting and terminal ones) has more than one ball contact. For simplicity, we assume the trajectory of a ball moving from one player to the next one is a line segment.

Lizarb's national soccer team observed that for some locations of kickers the two-ball game is possible but for some others it is impossible. The figure below shows two example locations: to the left, playing two-ball game is impossible; to the right, playing the game is possible.



Your task is to write a program that checks if for given player locations the two-ball game is possible or not.

Input

Each input starts with a single integer that gives the number of cases that follow. The firsts line of each case contains the number of players n, with $4 \leq n \leq 100\,000$, followed by n lines that describe the coordinates of the players. All coordinates are pairwise different and the points determined by the coordinates are not collinear (recall, three or more points are said to be collinear if they lie on a single straight line). The first coordinate describes the location of s_1, the second the location of t_1, the third coordinate describes the location of s_2, and the fourth the location of t_2. The remaining coordinates describe positions of other players of the team.

Output

For each case, your algorithm has to output a line containing "`POSSIBLE`" if it is possible to play the game and "`IMPOSSIBLE`", otherwise.

Sample input and output

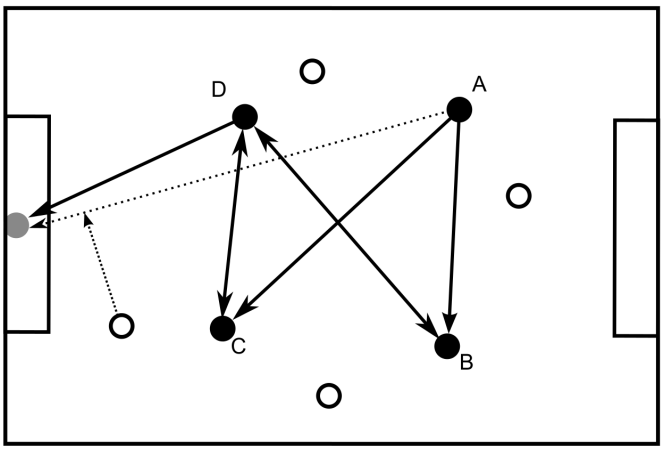
two.in	two.out
2 5 2.01 0.02 1.04 3.02 0.01 0.99 4.1 3.2 2.1 2.01 5 2.01 0.02 1.04 3.02 0.01 0.99 2.1 2.01 4.1 3.2	IMPOSSIBLE POSSIBLE

Problem K. To Score or Not to Score

Input file: `score.in`
Output file: `score.out`

Robot soccer matches in the very early days were quite funny, since most of the time there wasn't any action in the game. Robots only moved to catch a ball that a robot from the other team had shot. The reason for this somewhat strange behavior was attributed to the used strategy. The robots made a map of all players from the same team and the opponents. If one player was in possession of the ball before even shooting he tried to check whether it was possible to score from the current situation. In the process he checked if there was a way for the ball to reach the goal via several other players of his team. It was possible to shoot the ball to another player if no opponent was close enough to the shooting line to catch the ball. The opponent always moved perpendicular to the shooting line and only if he was sure that he could intercept the shot. The ball always traveled three times as fast as a robot could move, i.e. the player had to be quite near the shooting line to intercept the shot.

The other part of the game was fouling another player in order to prevent the other team from reaching the goal. The rules stated that only one player could be fouled at a time, so this only happened if fouling this one player prevented the other team from scoring. Also the initial shooter may not be fouled at any time. Fouling happens almost instantly by knocking the robot out with an electromagnetic pulse, thus the distance between the two opponents does not matter and the fouling robot does not move.



Let's have a look at the figure. There are four players in each team, indicated by the black and white circles. Suppose player A has the ball. Then only the shown shots are possible (note the directions). In all other cases a white player is near enough to intercept the shot (e.g. the direct shot from A to the goal). Thus in principle the black team could score. However, if player D is fouled the goal can no longer be scored.

Your university has decided to program a simulator for these ancient robot football matches and your task is to write the part that checks if the team in possession of the ball is able to score or not, based on the current positions of all robots. As a summary of the description above, a team may score if the ball cannot be intercepted by an opponent player and if more than one player would have to be fouled.

Input

The first line contains the number of testcases $1 \leq k \leq 10$ that follow. The first line of each testcase holds the number n of players per team ($1 \leq n \leq 20$). The next $2n$ lines contain the coordinates of all players, the first n lines being the first team, the second n lines the second team. A coordinate is given as two non-negative floating point numbers separated by spaces. The first player of the first team is in possession of the ball. The coordinates of the goal follow below the two teams. You may assume that the inputs are chosen in such a way that small floating point errors do not lead to wrong results.

Output

Output "`Goal`" if the first team is able to score or "`No goal`" if the ball can be intercepted or if fouling one player is enough to prevent the team from scoring.

Sample input and output

score.in	score.out
2 4 407.89 297.33 396.64 80.21 190.26 96.43 210.73 290.67 345.43 315.24 462.45 218.22 291.76 60.82 104.98 113.45 0 191.18 4 407.89 297.33 396.64 80.21 190.26 96.43 210.73 290.67 521.43 369.86 565.14 368.22 563.25 328.18 521.31 334.00 0 191.18	No goal Goal