Problem A
Class Statistics

The new principal of *Woop Woop* Public plans to meet the teaching team to discuss the performance of the classes/teachers and, being a bean counting fundamentalist, he wants to arm himself with some statistics for the meetings.

Your task is to write a program that reads the pupils' marks in each class and generates performance reports for the principal prior to the meetings.

Input

The input starts with an integer K ($1 \leq K \leq 100$) indicating the number of classes on a line by itself. Each of the following K lines gives a class's data, which starts with an integer N ($2 \leq N \leq 50$) indicating the number of pupils in the class. The number of pupils is followed by their marks, given as integers, in the range of zero to one hundred, separated by single spaces.

Output

The report for each class consists of two lines.

- The first line consists of the sentence: "Class X", where X indicates the class number starting with the value of one.
- The second line reports the maximum class mark, minimum class mark and the largest difference between consecutive marks (when sorted in non-decreasing order) in the class using the formats shown in the sample below.

Sample Input	Output for the Sample Input
2 5 30 25 76 23 78 6 25 50 70 99 70 90	Class 1 Max 78, Min 23, Largest gap 46 Class 2 Max 99, Min 25, Largest gap 25

Problem B
Avoiding a disaster

Percy likes to be punctual. So much so that he always keeps three watches with him, so that he can be sure exactly what the time is. However, Percy's having a bad day. He found out that one of his watches was giving the wrong time. What's worse, when he went to correct the watch, he corrected the wrong one! That is, one watch was running x minutes behind (where $x \leq 480$) and he wound one of the other watches x minutes forward. He now has three watches reading three different times, and hence is in serious danger of being tardy. Can you help Percy by writing a program that takes in the three times displayed on the watches and returns the correct time?

Input

The input begins with an integer T indicating the number of cases that follow ($0 < T < 100$). Each of the following T lines contains one test case, made up of three readings, separated by single space characters: H1:M1 H2:M2 H3:M3 In each reading H1,H2,H3 represent the hours displayed ($0 < H1,H2,H3 < 13$), and M1,M2,M3 represent the minutes displayed ($0 \leq M1,M2,M3 < 60$).

If the number of minutes is less than 10, a leading 0 is perpended.

Output

For each test case, one line should be produced, formatted exactly as follows: "The correct time is Hi:Mi". If the number of minutes is less than 10, a leading 0 should be added. If the number of hours is less than 10, a leading 0 should NOT be added. If it is impossible to tell the time from the three readings, print the string: "Look at the sun".

Sample Input	Output for the Sample Input
3 5:00 12:00 10:00 11:59 12:30 1:01 12:00 4:00 8:00	The correct time is 5:00 The correct time is 12:30 Look at the sun

Problem C
Roll-call in *Woop Woop* High

The new principal of *Woop Woop* High is not satisfied with her pupils performance. She introduced a new roll-call process to get a daily measure of the pupils' learning, which proceeds as follows:

At the beginning of the daily roll-call period each pupil is handed a question, which they must attempt to answer, before proceeding to their classes. A pupil stops after the question is answered correctly. Each pupil is allowed up to five attempts to answer the question correctly.

Pupils who answer correctly on the first attempt are marked present. Pupils who answer correctly after more than one attempt are encouraged to work at home. Pupils who fail to develop a correct answer within five attempts are given remedial classes after school. Pupils who do not give any answer are marked as abscent.

Your task is to write a program that reads the pupils' assessments and generates performance reports for the principal to proceed with appropriate actions.

Input

The input starts with an integer K ($1 \leq K \leq 100$) indicating the number of classes. Each class starts with an integer N ($1 \leq N \leq 50$) indicating the number of pupils in the class. Each of the following N lines starts with a pupil's name followed by up-to five assessments of his/her answers. An assessment of 'yes' or 'y' indicates a correct answer and an assessment of 'n' or 'no' indicates a wrong answer. A pupil's name consists of a single string with no white spaces.

Output

The attendance report for each class consists of five lines.

- The first line consists of the sentence: "Roll-call: X", where X indicates the class number starting with the value of one.
- The second line consists of the sentence: "Present: Y1 out of N", where Y1 is the number of pupils who did not submit a wrong answer.
- The third line consists of the sentence: "Needs to study at home: Y2 out of N", where Y2 is the number of pupils who submitted a number of wrong answers before submitting the correct answer.
- The fourth line consists of the sentence: "Needs remedial work after school: Y3 out of N", where Y3 indicates the number of pupils whose submitted five wrong answers.
- The fifth line consists of the sentence: "Absent: Y4 out of N", where Y4 indicates the number of absent pupils.

Sample Input	Output for the Sample Input
2 5 Doc n y sneezy n n no yes princecharming no n no no n goofy yes grumpy n y 5 evilemperor n y princesslia r2d2 no no y obeyonecanopy n no y darthvedar y	Roll-call: 1 Present: 1 out of 5 Needs to study at home: 3 out of 5 Needs remedial work after school: 1 out of 5 Absent: 0 out of 5 Roll-call: 2 Present: 1 out of 5 Needs to study at home: 3 out of 5 Needs remedial work after school: 0 out of 5 Absent: 1 out of 5

acm International Collegiate Programming Contest 2011 IBM event sponsor

South Pacific Region

Problem D
Difficult Routes

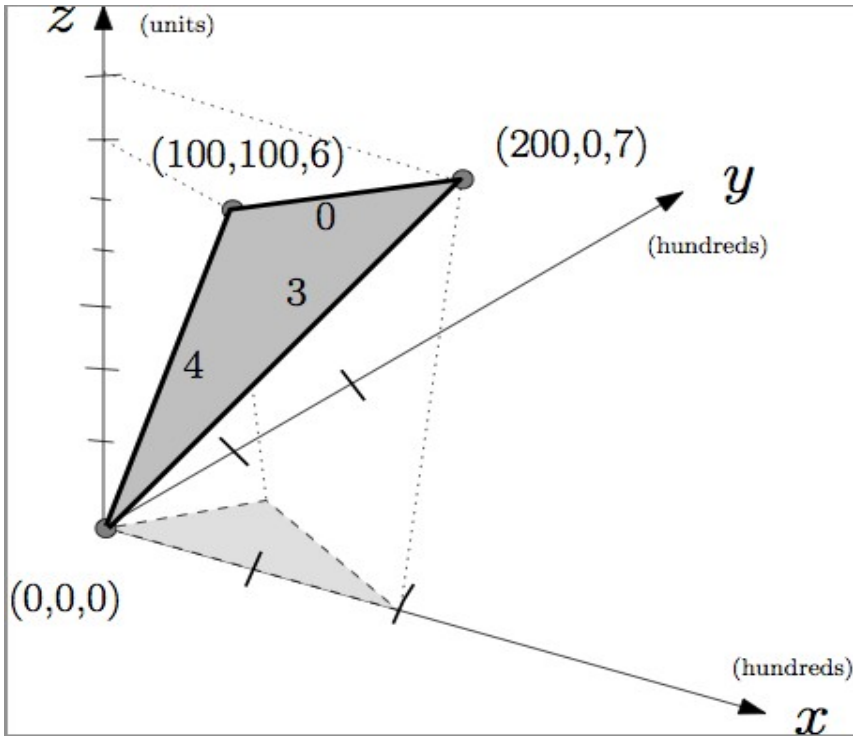
In preparation for the coming Olympics, you have been asked to propose bicycle training routes for your country's team. The training committee wants to identify routes for traveling between pairs of locations in multiple sites around the country. Each route must have a desired level of difficulty based on the steepness of its hills.

You will be given a road map with the elevation data superimposed upon it. Each intersection, where two or more roads meet, is identified by its x- , y- , and z-coordinates. Each road starts and ends at an intersection, is straight, and does not contain bridges over or tunnels under other roads. The difficulty level, *d*, of cycling a road is 0 if the road is level or travelled in the downhill direction. The difficulty of a non-level road when travelled in the uphill direction is $\lfloor 100 * rise / run \rfloor$. Here *rise* is the absolute value of change in elevation and *run* is the distance between its two intersection points in its horizontal projection to the 2D-plane at elevation zero. Note that the level of difficulty for cycling a descending road is zero.

A route, which is a sequence of roads such that a successor road continues from the same intersection where its predecessor road finishes, has a level of difficulty *d* if the maximum level of difficulty for cycling among all its roads equals *d*. The committee is also interested in the chosen route between two selected locations, if such a route with the desired difficulty level exists, being the one with the shortest possible distance to travel.

Reminder: The floor function $\lfloor X \rfloor$ means X truncated to an integer.

The figure shows a road map with three intersections for the three sample inputs.



The edge labels of the darker shaded surface give the level of difficulty of going up hill. The lighter shaded surface is the horizontal projection to the 2D-plane at elevation zero.

Input

Input consists of many road maps. Each map description begins with two non-negative integers N and M, separated by a space on a line by themselves, that represent the number of intersections and the number of roads, respectively. $0 <$ N, M ≤ 10000. A value of both N and M equal to zero denotes the end of input data.

Each of the next N lines contains three integers, separated by single spaces, which represent the x-, y- and z-coordinates of an intersection. The integers have values between 0 and 10000, inclusive. Intersections are numbered in order of their appearance starting with the value one. Each of the following M lines contains two integers that represent start and end intersections of a road.

Finally, three integers *s*, *t* and *d* that represents the desired starting intersection number *s*, the finishing intersection number *t* and the level of difficulty *d* for a training route are given on line by themselves. A valid training route must have at least one road with a difficulty level of *d*, and no road with a difficulty level greater than *d*. $0 \leq d \leq 10$. If the training route is meant to form a closed circuit, then *s* and *t* are the same intersection numbers.

Output

For each road map and desired route, the output consists of a single line that contains:

1. number denoting the shortest length of a training route rounded to three decimal places (where trailing zeros may be omitted), or
2. the single word “None” if no feasible route exists.

Reminder: Rounding a positive number R.xxxy to three decimal places

- If the fourth decimal place is less than 5, then the rounded value is R.xxx
- Otherwise, the rounded value is R.xxx + 0.001

Examples are: for the value of 10.3463 the output should be 10.346, and for the value of 10.3695 the output is either 10.37 or 10.370

Sample Input	Output for the Sample Input
3 3 0 0 0 100 100 6 200 0 7 1 2 2 3 3 1 1 2 3 3 3 0 0 0 100 100 6 200 0 7 1 2 2 3 3 1 1 1 4 3 3 0 0 0 100 100 6 200 0 7 1 2 2 3 3 1 2 1 5 0 0	341.547 283.097 None

Problem E
Real*Phobia*

Bert is a programmer with a real fear of floating point arithmetic. Bert has quite successfully used rational numbers to write his programs but he does not like it when the denominator grows large.

Your task is to help Bert by writing a program that decreases the denominator of a rational number, whilst introducing the smallest error possible. For a rational number A/B, where B > 2 and 0 < A < B, your program needs to identify a rational number C/D such that:

1. 0 < C < D < B, and
2. the error |A/B - C/D| is the minimum over all possible values of C and D, and
3. D is the smallest such positive integer.

Input

The input starts with an integer K ($1 \leq K \leq 1000$) that represents the number of cases on a line by itself. Each of the following K lines describes one of the cases and consists of a fraction formatted as two integers, A and B, separated by "/" such that:

1. B is a 32 bit integer strictly greater than 2, and
2. 0 < A < B

Output

For each case, the output consists of a fraction on a line by itself. The fraction should be formatted as two integers separated by "/".

Sample Input	Output for the Sample Input
3 1/4 2/3 13/21	1/3 1/2 8/13

Problem F
Insidious Branding

"I know a trick worth two of that"
-- William Shakespeare, Henry IV, Part 1, Act II, Scene 1

A brand designer came up with a strategy that you think will be the key to the success of your company, and therefore, you. The strategy is to use a brand name that has two possible decompositions into a pair of frequently used everyday words, and then expose the consumer to the four words repeatedly in brief episodes of sensory bombardment. The feeble mind of the consumer will jar, and the brand will then ease its way into the victim's long-term memory.

Your task is to write a program that sifts through a dictionary for combinations of four words (not necessarily distinct), say A, B, C and D, which satisfy the equation

$$A + B = C + D$$

where '+' denotes concatenation, '=' denotes an exact string match, the length of the word A is strictly less than the length of the word C, and all four strings are not empty.

Input

The input consists of multiple cases where one dictionary is used for each case. A case starts with an integer W ($1 \leq W < 100{,}000$) that indicates the number of words in the dictionary on a line by itself. Each of the following W lines contains a single word. The words are in no particular order, and they are all distinct within a test case. Each word is a string that consists of L lowercase letters and contains no spaces. $1 \leq L \leq 30$. The input is terminated by a zero on a line by itself.

Output

The output consists of a single integer, on a line by itself, that represents the number of equations that can be formed.

Sample Input	Output for the Sample Input
4 catchment ally catch mentally 0	1

Problem G
Help-or-else

A penal colony for finance professionals will soon be holding its annual community service activity with some rules that are considered suitable for a penal colony. Every inmate is assigned a set P of N people to help with their finances and a limit of *K* minutes. In addition to the circumstances of the j*th* person, $1 \leq j \leq N$, a time penalty of e_j for choosing not to give advice and the time duration of d_j minutes allotted to provide the advice are also made clear to the inmate.

An inmate starts his community service at time T equal to zero. If the inmate started working with the j*th* person at time T, then he must terminate his work no later than $T+d_j$. Regardless of the validity of the advice and time of completion, a value of C_j ($= T+ d_j$) is deducted from the inmate's alloted minutes. Also the inmate is not permitted to work with another person until the time $T+ d_j$.

If *S* is the set of people helped by an inmate, then the total number of used minutes is calculated as $\sum_{x \in S} C_x + \sum_{x \in P-S} e_x$.

Your task is to write a program to calculate the maximum number of persons that can be helped by an inmate without exceeding his *K* minutes limit.

Input

Input consists of sets for many inmates. The description for each inmate begins with two integers N and K, separated by a single space on a line by themselves, that represent the number of people and the maximum allowed minutes. $0 < N \leq 200$ and $0 < K \leq 6000$. Each of the following N lines contains two integers, separated by a single space, which represent the penalty and time duration one person to be assisted. All integers have values between 0 and 10000, inclusive. Input terminates with two zeros on a line by themselves.

Output

For each inmate, the output consists of a single line that contains the maximum number of persons to be helped within the given time limit using the format shown. "Mission Impossible" is entered where not exceeding the given time limit is not possible.

Sample Input	Output for the Sample Input
1 1000 100 1000 2 100 1000 1000 20 10 1 1 0 10000 4 293 61 30 295 39 206 27 94 85 0 0	1: 1 2: Mission Impossible 3: 0 4: 3

Problem H
Pahom on Water

Pahom on Water is an interactive computer game inspired by a short story of Leo Tolstoy about a poor man who, in his lust for land, forfeits everything.

The **game's starting screen** displays a number of circular pads painted with colours from the visible light spectrum. More than one pad may be painted with the same colour (defined by a certain frequency) except for the two colours red and violet. The display contains only one red pad (the lowest frequency of 400 THz) and one violet pad (the highest frequency of 789 THz). A pad may intersect, or even contain another pad with a different colour but never merely touch its boundary. The display also shows a figure representing *Pahom* standing on the red pad.

The **game's objective** is to walk the figure of *Pahom* from the red pad to the violet pad and return back to the red pad. The walk must observe the following rules:

1. If pad α and pad β have a common intersection and the frequency of the colour of pad α is strictly smaller than the frequency of the colour of pad β, then *Pahom* figure can walk from α to β during the walk from the red pad to the violet pad
2. If pad α and pad β have a common intersection and the frequency of the colour of pad α is strictly greater than the frequency of the colour of pad β, then *Pahom* figure can walk from α to β during the walk from the violet pad to the red pad
3. A coloured pad, with the exception of the red pad, disappears from display when the *Pahom* figure walks away from it.

The developer of the game has programmed all the whizzbang features of the game. All that is left is to ensure that *Pahom* has a chance to succeed in each instance of the game (that is, there is at least one valid walk from the red pad to the violet pad and then back again to the red pad.) Your task is to write a program to check whether at least one valid path exists in each instance of the game.

Input

The input starts with an integer K ($1 \leq K \leq 50$) indicating the number of scenarios on a line by itself. The description for each scenario starts with an integer N ($2 \leq N \leq 300$) indicating the number of pads, on a line by itself, followed by N lines that describe the colors, locations and sizes of the N pads. Each line contains the frequency, followed by the x- and y-coordinates of the pad's center and then the radius. The frequency is given as a real value with no more than three decimal places. The coordinates and radius are given, in meters, as integers.

All values are separated by a single space. All integer values are in the range of -10,000 to 10,000 inclusive.

In each scenario, all frequencies are in the range of 400.0 to 789.0 inclusive. Exactly one pad will have a frequency of "400.0" and exactly one pad will have a frequency of "789.0".

Output

The output for each scenario consists of a single line that contains:

- Game is VALID, or
- Game is NOT VALID

Sample Input	Output for the Sample Input
2 2 400.0 0 0 4 789.0 7 0 2 4 400.0 0 0 4 789.0 7 0 2 500.35 5 0 2 500.32 5 0 3	Game is NOT VALID Game is VALID

Problem I
Detour Buster

The Green Parcel Services company employs cyclists to deliver parcels across a large metropolitan city, and employees are paid according to the distances they must travel to deliver parcels. Each company bicycle has been equipped with a global positioning system that records its position once every couple of seconds. The sequence of positions for one delivery is called a track and the length of a track, which is the sum of euclidean distances between consecutive pairs of recorded points, is used to calculate the employee's payment.

A recent audit of the recorded tracks revealed that some tracks self-intersect, which indicates that some employees are making unnecessary detours. Your task is to write a program to process a given track and calculate the length of the shortest possible track from the first to the last point of the original track. The shortest possible track must be part of the original track and may include travelling in the same or opposite direction along the original track.

Input

Input starts with an integer, on a line by itself, that represents the number of tracks to be shortened. The description of each track begins with a positive integer N, on a line by itself, representing the number of recorded points that define the track. Each of the following N lines contains two integers, separated by a single space, that define the x- and y-coordinates of a point on the track in metres. The points are listed in order of their appearance on the track.

Two successive points, which form a segment, will be no more than thirty metres apart and a segment will not intersect more than twenty other segments. All x- and y-coordinates have values between -10,000,000 and 10,000,000, inclusive. N has values between 1 and 100,000, inclusive.

Output

For each input track, the output consists of a single integer, on a line by itself, which is the length of the shortest track in metres. The answer must be rounded to the nearest integer value.

Reminder: Rounding a positive number R.xyz to the nearest integer

- If the first decimal place (that is **x**) is less than 5, then the rounded value is R
- Otherwise, the rounded value is R+1

Sample Input	Output for the Sample Input
2 5 0 0 12 0 20 0 10 10 10 -10 6 0 0 15 0 10 -5 4 1 10 1 10 -10	20 17

Problem J
Oil Skimming

Thanks to a certain "green" resources company, there is a new profitable industry of oil skimming. There are large slicks of crude oil floating in the Gulf of Mexico just waiting to be scooped up by enterprising oil barons. One such oil baron has a special plane that can skim the surface of the water collecting oil on the water's surface. However, each scoop covers a 10m by 20m rectangle (going either east/west or north/south). It also requires that the rectangle be completely covered in oil, otherwise the product is contaminated by pure ocean water and thus unprofitable!

Given a map of an oil slick, the oil baron would like you to compute the maximum number of scoops that may be extracted. The map is an NxN grid where each cell represents a 10m square of water, and each cell is marked as either being covered in oil or pure water.

Input

The input starts with an integer K ($1 \leq K \leq 100$) indicating the number of cases. Each case starts with an integer N ($1 \leq N \leq 600$) indicating the size of the square grid. Each of the following N lines contains N characters that represent the cells of a row in the grid. A character of '#' represents an oily cell, and a character of '.' represents a pure water cell.

Output

For each case, one line should be produced, formatted exactly as follows: "Case X: M" where X is the case number (starting from 1) and M is the maximum number of scoops of oil that may be extracted.

Sample Input	Output for the Sample Input
1 6 ..... .##... .##... ....#. ....##	Case 1: 3

Problem K
Numbers Game

The objective of the numbers game is to use the basic arithmetic operations (+, -, * and /) and a number of given integers (4 to 7 integers) to get as close as possible to a given target integer. Each of the operations can be used multiple times, but each of the integers can be used at-most once. A player wins if he/she manages to calculate the closest possible value to the target integer. For example, the closest possible value to a target of 20 using the integers {2,3,5} can be achieved by the expression (2+5) * 3.

Your task is to write a program to calculate the answer for each game. Note that "/" stands for integer division (examples are: 5/4 equals 1, 8/9 equals 0.)

Input

The input consists of many games. The description for each game is given on two lines. The first line contains two integers T and N, separated by a single space, that represent the target integer and the number of given integers. $-700{,}000 \leq T \leq 700{,}000$ and $4 \leq N \leq 7$. The second line contains N integers separated by single spaces. All integers have values between -1,000,000 and 1,000,000 inclusive.

Two zeros on a line by themselves, separated by a single space, terminate the input.

Output

For each game, the output consists of a single line that contains the closest possible integer to the target. Any answer with the smallest distance to the target is correct.

Sample Input	Output for the Sample Input
30 5 1 2 3 4 5 10000 5 11 2 3 7 5 0 0	30 2310