Problem A
All about that base

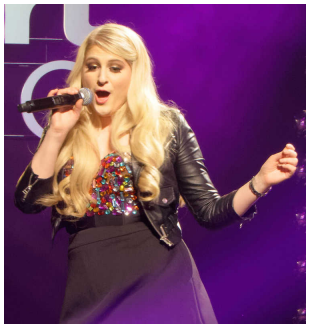


Photo by Ronald Woan

The *base* (or *radix*) of a positional numeral system is the number of symbols that can be used to represent a number in that system. The base 10 system (also known as decimal) uses 10 distinct symbols: $0, 1, \ldots, 9$. For example, we interpret the number 72345 as:

$$7 \times 10^4 + 2 \times 10^3 + 3 \times 10^2 + 4 \times 10^1 + 5 \times 10^0.$$

This example illustrates that in base 10 the symbol at place $P \geq 0$ (starting from the right) is multiplied by 10^P to get its value. More generally, in base B we use B symbols to represent $0, \ldots, B-1$, and the symbol at the Pth place is multiplied by B^P to get its value.

Other bases commonly used in computation include base 2 (or binary, using symbols 0 and 1), base 8 (or octal, using symbols 0–7), and base 16 (or hexadecimal, using symbols 0–9 and a–f). In bases higher than 10, letters represent the higher values. Thus in hexadecimal a–f represent the decimal values 10–15, and in bases ≥ 36 the letter z represents the decimal value 35.

Your job is to determine the bases in which given arithmetic expressions are valid. We define an expression as *valid* in base B if two conditions are true. First, all the operands used are interpretable in base B as having values in the decimal range $[1, 2^{32}-1]$. Second, the expression is true. Any arbitrary expression might be valid in zero, one, or more bases. In this problem we will only consider bases 1–36, where base 1 is unary.

Note that following the convention listed above, unary would consist of a single symbol: 0. In this problem, unary numbers use the symbol 1 rather than 0 (think "tally marks"). E.g., 111 in unary is equivalent to the decimal number 3 and 1111111 in unary is equivalent to the decimal number 7.

Input

Input for this problem starts with a line containing an integer $0 \leq N \leq 20$. The following N lines each contain an arithmetic expression with the following form:

$$X \; op \; Y \; = \; Z$$

where X, Y, and Z are positive, whole numbers consisting of 1 to 100 symbols from the set 0–9 and a–z, and op is one of the four operators `+`, `-`, `*`, `/`. For each statement there is at least one base $1 \leq B \leq 36$ such that X, Y, and Z can all be interpreted in base B as having values in the decimal range $[1, 2^{32}-1]$.

Output

For each expression, list the bases in which the expression is valid (sorted in ascending base order) or the word "invalid" if the expression not valid in any of the bases 1–36. Use symbols 1–9, then a–z, then 0 to represent bases 1–36 (with the last symbol, 0, representing base 36).

Sample Input 1	Sample Output 1
8 6ef + d1 = 7c0 3 / 2 = 1 444 / 2 = 222 10111 * 11 = 1000101 10111 * 11 = 111221 5k - 1z = 46 1111111111 - 1111111 = 111 2048 - 512 = 1536	g invalid 56789abcdefghijklmnopqrstuvwxyz0 2 3456789abcdefghijklmnopqrstuvwxyz0 invalid 1 a

Problem B
Bobby's Bet

Bobby and Betty have a bet. Betty bets Bobby that he cannot roll an S-sided die (having values 1 through S) and obtain a value $\geq R$ on at least X out of Y rolls. Betty has a variety of dice with different numbers of sides S, and all her dice are fair (for a given die, each side's outcome is equally likely). In order to observe statistically rare events while still giving Bobby a reason to bet, Betty offers to pay Bobby W times his bet on each encounter. For example, suppose Betty bets Bobby 1 bitcoin that he can't roll at least a 5 on a 6-sided die at least two out of three times; if Bobby does, she would give him $W = 3$ times his initial bet (i.e. she would give him 3 bitcoins). Should Bobby take the bet (is his expected return greater than his original bet)?

Photo by StarsApart

Input

Input begins with an integer $1 \leq N \leq 10\,000$, representing the number of cases that follow. The next N lines each contain five integers, R, S, X, Y, and W. Their limits are $1 \leq R \leq S \leq 20$, $1 \leq X \leq Y \leq 10$, and $1 \leq W \leq 100$.

Output

For each case, output "yes" if Bobby's expected return is greater than his bet, or "no" otherwise. Bobby is somewhat risk averse and does not bet if his expected return is equal to his bet.

Sample Input 1	Sample Output 1
2 5 6 2 3 3 5 6 2 3 4	no yes

Sample Input 2	Sample Output 2
3 2 2 9 10 100 1 2 10 10 1 1 2 10 10 2	yes no yes

This page is intentionally left blank.

Problem C
Cantina of Babel

Characters in Star Wars each speak a language, but they typically understand a lot more languages that they don't or can't speak. For example, Han Solo might speak in Galactic Basic and Chewbacca might respond in Shyriiwook; since they each understand the language spoken by the other, they can communicate just fine like this.

Photo by Brickset

We'll say two characters can *converse* if they can exchange messages in both directions. Even if they didn't understand each other's languages, two characters can still converse as long as there is a sequence of characters who could translate for them through a sequence of intermediate languages. For example, Jabba the Hutt and R2D2 might be able to converse with some help. Maybe when Jabba spoke in Huttese, Boba Fett could translate to Basic, which R2D2 understands. When R2D2 replies in Binary, maybe Luke could translate to Basic and then Bib Fortuna could translate back to Huttese for Jabba.

In Star Wars Episode IV, there's a scene with a lot of different characters in a cantina, all speaking different languages. Some pairs of characters may not be able to converse (even if others in the cantina are willing to serve as translators). This can lead to all kinds of problems, fights, questions over who shot first, etc. You're going to help by asking some of the patrons to leave. The cantina is a business, so you'd like to ask as few as possible to leave. You need to determine the size of the smallest set of characters S such that if all the characters in S leave, all pairs of remaining characters can converse.

For example, in the first sample input below, Chewbacca and Grakchawwaa can converse, but nobody else understands Shyriiwook, so they can't converse with others in the bar. If they leave, everyone else can converse. In the second sample input, Fran and Ian can converse, as can Polly and Spencer, but no other pairs of characters can converse, so either everyone but Polly and Spencer must leave or everyone but Fran and Ian.

Input

Input starts with a positive integer, $1 \leq N \leq 100$, the number of characters in the cantina. This is followed by N lines, each line describing a character. Each of these N lines starts with the character's name (which is distinct), then the language that character speaks, then a list of 0 to 20 additional languages the character understands but doesn't speak. All characters understand the language they speak. All character and language names are sequences of 1 to 15 letters (a-z and A-Z), numbers, and hyphens. Character names and languages are separated by single spaces.

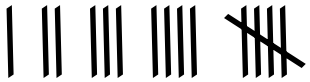
Output

Print a line of output giving the size of the smallest set of characters S that should be asked to leave so that all remaining pairs of characters can converse.

Sample Input 1	Sample Output 1
7 Jabba-the-Hutt Huttese Bib-Fortuna Huttese Basic Boba-Fett Basic Huttese Chewbacca Shyriiwook Basic Luke Basic Jawaese Binary Grakchawwaa Shyriiwook Basic Jawaese R2D2 Binary Basic	2

Sample Input 2	Sample Output 2
6 Fran French Italian Enid English German George German Italian Ian Italian French Spanish Spencer Spanish Portugese Polly Portugese Spanish	4

Problem D
Circuit Counting



Suppose you are given a sequence of N integer-valued vectors in the plane (x_i, y_i), $i = 1, \ldots, N$. Beginning at the origin, we can generate a path by regarding each vector as a displacement from the previous location. For instance, the vectors $(1, 2)$, $(2, 3)$, $(-3, -5)$ form the path $(0, 0), (1, 2), (3, 5), (0, 0)$. We define a path that ends at the origin as a *circuit*. The example just given is a circuit. We could form a path using any nonempty subset of the N vectors, while the result (circuit or not) doesn't depend on the ordering of the subset. How many nonempty subsets of the vectors form circuits?

For instance, consider the vectors $\{(1, 2), (-1, -2), (1, 1), (-2, -3), (-1, -1)\}$ From these vectors we can construct 4 possible subset circuits using

$$\{(1, 2),\ (-1, -2)\}$$
$$\{(1, 1),\ (-1, -1)\}$$
$$\{(1, 2),\ (1, 1),\ (-2, -3)\}$$
$$\{(1, 2),\ (-1, -2),\ (1, 1),\ (-1, -1)\}$$

Input

Input begins with an integer $N \leq 40$ on the first line. The next N lines each contain two integer values x and y forming the vector (x, y), where $|x|, |y| \leq 10$ and $(x, y) \neq (0, 0)$. Since the given vectors are a set, all vectors are unique.

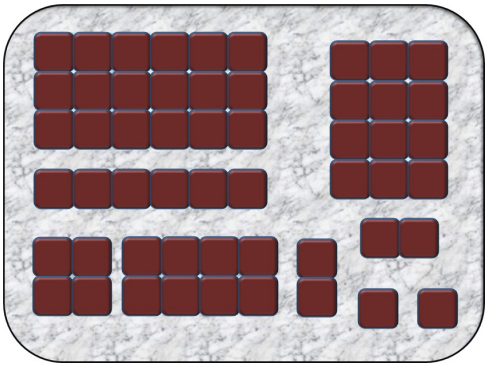
Output

Output the number of nonempty subsets of the given vectors that produce circuits. It's guaranteed that the answer is less than 10^{10}.

Sample Input 1	Sample Output 1
5 1 2 1 1 -1 -2 -2 -3 -1 -1	4

This page is intentionally left blank.

Problem E
Cutting Brownies



John Horton Conway (1937-) is a British mathematician with many contributions to mathematics. He is famous for the invention of the cellular automaton, more popularly known as the "Game of Life." This problem is inspired by a game Conway invented in the 1970s.

This game is played using a rectangular sheet of brownies fresh out of the oven. The players are Harry Horizontal and Vicky Vertical. Initially, there is a single piece consisting of $B \times D$ connected squares (the individual brownies).

At each turn, a player chooses one of the remaining pieces and if possible, cuts it into two smaller pieces such that both pieces have integer breadth and depth. Harry may make only horizontal cuts, Vicky only vertical cuts. Pieces may not be rotated before or after a cut. If a player cannot cut any of the remaining pieces, that player loses.

Let's consider some examples. The simplest game is  . In this case, neither Harry nor Vicky can make a move, so whoever starts loses. On the other hand,  is a win for Harry, no matter who starts. Similarly,  is a win for Vicky, no matter who starts.

Consider  , which is a loss for whoever starts. For instance, if Vicky starts, her only move leaves Harry with  ,  and once he cuts any of the pieces, Vicky is left with  ,  ,  (in any order) and thus again without moves. For reasons of symmetry, Harry loses if he is made to start.

Intuition might tell us that Vicky should tend to win if the initial sheet is broader than it is deep (since such sheets yield more opportunities for vertical cuts), but consider  . If Harry starts, his only possible move leaves Vicky with  ,  and a win. But if Vicky starts, any possible move leaves Harry with  ,  . Harry responds and leaves Vicky with  ,  ,  , which Vicky will eventually lose since there are no moves left in the 2 sheets and whoever makes the first move on  loses.

On the other hand,  is a winner for Vicky, no matter who starts. If Harry starts, he runs out of moves after his first cut. If Vicky starts, her best move is to cut in the center, leaving Harry with  ,  , which he loses because each  game is lost by whoever moves first.

Given the initial size of the sheet, and given who starts the game, write a program that computes if the starting player has a strategy to force a win!

Input

The first line contains an integer $1 \leq N \leq 10$ denoting the number of test cases that follow. Each test case consists of a single line containing two integers B and D, and a string S. Here B denotes the initial breadth of the sheet ($1 \leq B \leq 500$), D denotes the initial depth of the sheet ($1 \leq D \leq 500$) and S is either `Harry` or `Vicky` depending on whether Harry or Vicky moves first.

Output

For each test case, output whether the player who starts can force a win in the game. Output the player's name followed by `can win` or `cannot win`.

Sample Input 1	Sample Output 1
5 1 1 Harry 2 2 Vicky 3 2 Vicky 4 2 Vicky 6 8 Harry	Harry cannot win Vicky cannot win Vicky cannot win Vicky can win Harry can win

Problem F
Quick Brown Fox

A *pangram* is a phrase that includes at least one occurrence of each of the 26 letters, 'a'...'z'. You're probably familiar with this one: "The quick brown fox jumps over the lazy dog."

Your job is to recognize pangrams. For phrases that don't contain every letter, report what letters are missing. We'll say that a particular letter occurs in the phrase if it occurs as either upper case or lower case.

Photo by Neil McIntosh

Input

Input starts with a line containing an integer $1 \leq N \leq 50$. The next N lines are each a single phrase, possibly containing upper and lower case letters, spaces, decimal digits and punctuation characters '.', ',', '?', '!', ''' and '"'. Each phrase contains at least one and no more than 100 characters.

Output

For each input phrase, output "pangram" if it qualifies as a pangram. Otherwise, output the word "missing" followed by a space and then the list of letters that didn't occur in the phrase. The list of missing letters should be reported in lower case and should be sorted alphabetically.

Sample Input 1

```
3
The quick brown fox jumps over the lazy dog.
ZYXW, vu TSR Ponm lkj ihgfd CBA.
.,?!'" 92384 abcde FGHIJ
```

Sample Output 1

```
pangram
missing eq
missing klmnopqrstuvwxyz
```

This page is intentionally left blank.

Problem G
Safe Passage

A group of friends snuck away from their school campus, but now they must return from the main campus gate to their dorm while remaining undetected by the many teachers who patrol the campus. Fortunately, they have an invisibility cloak, but it is only large enough to cover two people at a time. They will take turns as individuals or pairs traveling across campus under the cloak (and by necessity, returning the cloak to the gate if others remain). Each student has a maximum pace at which he or she is able to travel, yet if a pair of students are walking under the cloak together, they will have to travel at the pace of the slower of the two. Their goal is to have everyone back at the dorm as quickly as possible.

Photo by Ian Burt

As an example, assume that there are four people in the group, with person A able to make the trip in 1 minute, person B able to travel in 2 minutes, person C able to travel in 7 minutes, and person D able to travel in 10 minutes. It is possible to get everyone to the dorm in 17 minutes with the following plan:

– A and B go from the gate to the dorm together	(taking 2 minutes)
– A returns with the cloak to the gate	(taking 1 minute)
– C and D go from the gate to the dorm together	(taking 10 minutes)
– B returns with the cloak to the gate	(taking 2 minutes)
– A and B go from the gate to the dorm together	(taking 2 minutes)

Input

The input is a single line beginning with an integer, $2 \leq N \leq 15$. Following that are N positive integers that respectively represent the minimum time in which each person is able to cross the campus if alone; these times are measured in minutes, with each being at most $5\,000$. (It is a very large campus!)

Output

Output the minimum possible time it takes to get the entire group from the gate to the dorm.

Sample Input 1	Sample Output 1
2 15 5	15

Sample Input 2	Sample Output 2
4 1 2 7 10	17

Sample Input 3	Sample Output 3
5 12 1 3 8 6	29

This page is intentionally left blank.

Problem H
Secret Message

Jack and Jill developed a special encryption method, so they can enjoy conversations without worrrying about eavesdroppers. Here is how: let L be the length of the original message, and M be the smallest square number greater than or equal to L. Add $(M - L)$ asterisks to the message, giving a padded message with length M. Use the padded message to fill a table of size $K \times K$, where $K^2 = M$. Fill the table in row-major order (top to bottom row, left to right column in each row). Rotate the table 90 degrees clockwise. The encrypted message comes from reading the message in row-major order from the rotated table, omitting any asterisks.

For example, given the original message 'iloveyouJack', the message length is $L = 12$. Thus the padded message is 'iloveyouJack****', with length $M = 16$. Below are the two tables before and after rotation.

i	l	o	v
e	y	o	u
J	a	c	k
*	*	*	*

*	J	e	i
*	a	y	l
*	c	o	o
*	k	u	v

Then we read the secret message as 'Jeiaylcookuv'.

Input

The first line of input is the number of original messages, $1 \leq N \leq 100$. The following N lines each have a message to encrypt. Each message contains only characters a–z (lower and upper case), and has length $1 \leq L \leq 10\,000$.

Output

For each original message, output the secret message.

Sample Input 1	Sample Output 1
2 iloveyoutooJill TheContestisOver	iteiloyllooоJuv OsoTvtnheiterseC

This page is intentionally left blank.

Problem I
Simon Says

Photo by David Amsler

In the game "Simon Says" one person plays the role of Simon, who gives instructions to everyone else playing the game. The tricky part is that if Simon begins his instruction with "Simon says" then everyone else *must* follow the instruction (or they lose the game); if Simon gives an instruction that does not begin with "Simon says" then everyone is supposed to completely ignore the instruction (or they lose the game)!

Simon tries his or her best to trick the other players into following the wrong instructions. Simon might begin by saying "Simon says touch your nose." and follow this with "Stop touching your nose." Anyone who stops touching their nose loses! The last player still remaining, who has correctly followed precisely the instructions that began with "Simon says" (and only these instructions), gets to be Simon next.

As a child, you were horrible at this game. Your older siblings were always able to trick you into following the wrong instructions. Well, you will have the last laugh: now that you are a computer programmer, you can write a computer program that can help you play the game perfectly. You only need to make sure the program is able to determine which instructions to follow and which to ignore.

Are you up to the challenge? Can you craft a computer program that *never* makes any mistakes in the game? If you can, then surely fame and glory shall come your way for being the most unstoppable player of Simon Says ever!

Input

Input starts with a line containing an integer $1 \leq N \leq 1\,000$. Each of the next N lines is one command, of length at most 100 characters. Each command is a properly-capitalized sequence of one or more words, separated by a single space between each pair of words, ending in a period. Some commands begin with "Simon says" and others may not. If a command begins with "Simon says", there will always be another space and at least one additional word after "says". No lines contain leading or trailing space.

Output

For each line that begins with precisely "Simon says", output the rest of the line. Each line that does not begin with precisely "Simon says" should be ignored.

Sample Input 1	Sample Output 1
1 Simon says smile.	smile.

Sample Input 2

```
3
Simon says raise your right hand.
Lower your right hand.
Simon says raise your left hand.
```

Sample Output 2

```
raise your right hand.
raise your left hand.
```

Sample Input 3

```
3
Raise your right hand.
Lower your right hand.
Simon says raise your left hand.
```

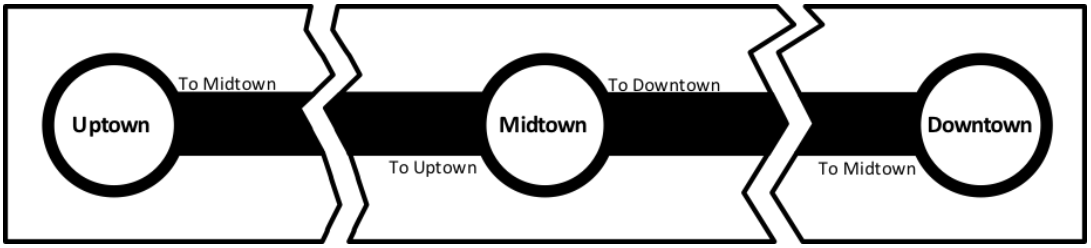
Sample Output 3

```
raise your left hand.
```

Problem J
Torn To Pieces

You have arrived in The Big City but your journey is not yet complete. You must still navigate the subway and get to your final destination. The information booth in the subway station is unattended and fresh out of maps of the subway system. On the floor you notice fragments of a map. Can you piece together enough of the map to figure out how to get to your final destination?

Each fragment of the map happens to perfectly contain a single subway station while also identifying all of the other stations that it connects to. Each connection between stations is bi-directional such that it can be travelled going either direction. Using all of the available fragments, your task is to determine the sequence of stations you must pass through in order to reach your final destination or state that there is no route if you don't have enough information to complete your journey.



Input

The first line of input has an integer, $2 \leq N \leq 32$, that identifies the number of pieces of the map that were found.

The following N lines each describe a station depicted on one of those pieces. Each of these lines starts with the name of the station they describe and is followed by a space-separated list of all of the station names that are directly connected to that station (there may be as many as $N - 1$).

The final line identifies a starting station and a destination station. The destination station is guaranteed to be different than the starting station.

Each station name is a string of up to 20 characters using only letters a–z and A–Z. It is guaranteed that there is at most one simple route (without revisiting stations) from the starting station to the destination station.

Output

Give the sequence of stations that leads from the starting station to the destination station. Separate station names with spaces. If there are not enough pieces of the map to find a route from the starting station to the destination station then output "`no route found`".

Sample Input 1	Sample Output 1
3 Uptown Midtown Midtown Uptown Downtown Downtown Midtown Uptown Downtown	Uptown Midtown Downtown

Sample Input 2	Sample Output 2
6 A B B A D C D E D F G F E G E F A	F E D B A

Sample Input 3	Sample Output 3
4 FirstStop SecondStop SecondStop FirstStop ThirdStop FifthStop FourthStop SixthStop SixthStop FifthStop FirstStop FifthStop	no route found

Problem K
UnDetected

The Department of Defense has been designing autonomous robots that can infiltrate war zones and other hostile places in order to carry out missions. Now they want to test their latest design, the Penetrator1700, and they've hired you to help design the test environment.

The test environment is a rectangular field with some sensors placed within the field. Each sensor has a certain radius defining the region within which it can detect a robot. You want to design the field to have as many sensors as possible while still permitting a route across the field that avoids detection.

The field is a region of the coordinate plane defined by $0 \leq x \leq 200$ and $0 \leq y \leq 300$. The robot can be modeled by a point that must remain on the field at all times. It starts at the bottom of the field ($y = 0$) and must end at the top of the field ($y = 300$), and must not pass within range of any sensor. There are N sensor locations given by triples (x, y, r) of integers, where each (x, y) is a point on the field, and r is its radius of detection. The implied sensor circles may overlap, but will never be tangent with each other nor with the boundary of the field. All sensors are initially inactive. You must find the largest value of k such that if sensors $1, 2, 3, \ldots, k$ are activated there is a path for the robot across the field, but no path if the (k+1)st sensor is also activated. It is guaranteed that there is no path if all N sensors are activated.

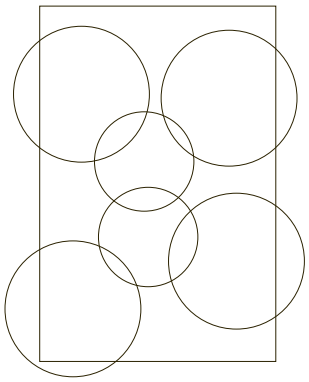


Figure K.1: Sensor circles corresponding to the first three sample inputs.

Input

Input begins with a positive integer $N \leq 200$. Each of the next N lines has three space-separated integers, representing x, y, r for a sensor, where $r \leq 300$. All sensors lie at different (x, y) positions. The first three sample inputs below correspond to the figure shown.

Output

Output a single integer (which may be 0) giving the largest k as described above.

Sample Input 1	Sample Output 1
6 36 228 58 164 224 58 88 170 42 93 105 42 167 85 58 28 44 58	2

Sample Input 2	Sample Output 2
6 36 228 58 28 44 58 164 224 58 88 170 42 93 105 42 167 85 58	3

Sample Input 3	Sample Output 3
6 28 44 58 36 228 58 88 170 42 93 105 42 164 224 58 167 85 58	4

Sample Input 4	Sample Output 4
3 100 150 101 30 30 10 170 30 100	0