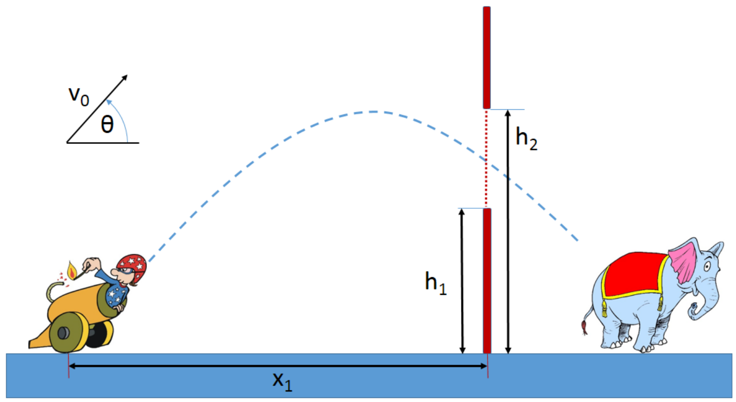
Problem A
The Amazing Human Cannonball
Time limit: 1 second



Source: picgifs.com

The amazing human cannonball show is coming to town, and you are asked to double-check their calculations to make sure no one gets injured! The human cannonball is fired from a cannon that is a distance x_1 from a vertical wall with a hole through which the cannonball must fly. The lower edge of the hole is at height h_1 and the upper edge is at height h_2. The initial velocity of the cannonball is given as v_0 and you also know the angle θ of the cannon relative to the ground.

Thanks to their innovative suits, human cannonballs can fly without air resistance, and thus their trajectory can be modeled using the following formulas:

$$x(t) = v_0 t \cos\theta$$
$$y(t) = v_0 t \sin\theta - \frac{1}{2} g t^2$$

where $x(t), y(t)$ provides the position of a cannon ball at time t that is fired from point $(0, 0)$. g is the acceleration due to gravity ($g = 9.81\ m/s^2$).

Write a program to determine if the human cannonball can make it safely through the hole in the wall. To pass safely, there has to be a vertical safety margin of $1m$ both below and above the point where the ball's trajectory crosses the centerline of the wall.

Input

The input will consist of up to 100 test cases. The first line contains an integer N, denoting the number of test cases that follow. Each test case has 5 parameters: v_0 θ x_1 h_1 h_2, separated by spaces. v_0 $(0 < v_0 \leq 200)$ represents the ball's initial velocity in m/s. θ is an angle given in degrees $(0 < \theta < 90)$, x_1 $(0 < x_1 < 1000)$ is the distance from the cannon to the wall, h_1 and h_2 $(0 < h_1 < h_2 < 1000)$ are the heights of the lower and upper edges of the wall. All numbers are floating point numbers.

Output

If the cannon ball can safely make it through the wall, output "`Safe`". Otherwise, output "`Not Safe`"!

Sample Input 1

```
11
19 45 20 9 12
20 45 20 9 12
25 45 20 9 12
20 43 20 9 12
20 47.5 20 9 12
20 45 17 9 12
20 45 24 9 12
20 45 20 10 12
20 45 20 9 11
20 45 20 9.0 11.5
20 45 18.1 9 12
```

Sample Output 1

```
Not Safe
Safe
Not Safe
Not Safe
Not Safe
Not Safe
Not Safe
Not Safe
Safe
Safe
```

Problem B
Recount
Time limit: 3 seconds

The recent schoolboard elections were hotly contested: a proposal to swap school start times for elementary and high school students, a controversial new dress code proposal that bans athletic clothes in school, and a proposal to raise real-estate taxes to pay for a new football practice facility, and the list goes on and on. It is now hours after the polls have closed and a winner has yet to emerge!

In their desperation, the election officials turn to you and ask you to write a program to count the vote!

Input

The input consists of a single test case, which is a list of votes cast. Each line in the input contains the name of a candidate for whom a vote was cast. A name may consist of multiple words, separated by spaces. Words contain letters or hyphens, but no other punctuation characters. There will be at least 2 votes on the list. The list of votes ends with a single line containing the characters `***`. This line should not be counted. There can be up to $100,000$ valid votes.

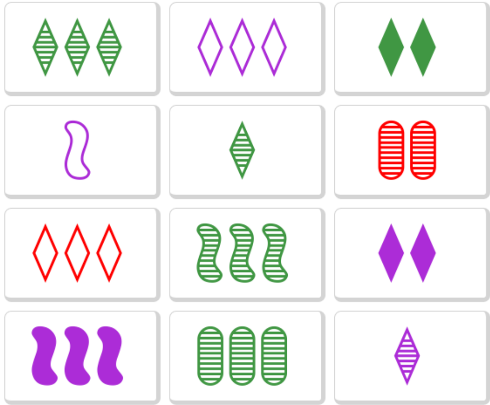
Output

If a candidate obtained a simple or absolute majority of all votes cast (that is, more than any other candidate), output the name of this candidate! If no candidate obtained a simple majority, output: "`Runoff!`" (don't forget to include the exclamation mark!)

Sample Input 1	Sample Output 1
Penny Franklin Marti Graham Connie Froggatt Joseph Ivers Connie Froggatt Penny Franklin Connie Froggatt Bruce Stanger Connie Froggatt Barbara Skinner Barbara Skinner ***	Connie Froggatt

Sample Input 2	**Sample Output 2**
Penny Franklin Connie Froggatt Barbara Skinner Connie Froggatt Jose Antonio Gomez-Iglesias Connie Froggatt Bruce Stanger Barbara Skinner Barbara Skinner ***	Runoff!

Problem C
Set!
Time limit: 1 second



SET is a card game designed by Marsha Falco in 1974 which is marketed by Set Enterprises, Inc. It also appears in syndicated form on the website of the New York Times. The player is shown 12 cards (see illustration), each of which contains 1, 2, or 3 symbols. The symbols are either diamonds, squiggles, or ovals. Symbols are drawn using either a solid, striped, or open fill style. Each symbol's color is either red, green, or purple. On a given card, all symbols are of the same type, same color, and have the same fill style.

To make a set, you must select three cards for which all 4 characteristics are either the same or pairwise different. For instance, 3 cards where the first shows 2 striped red ovals, the second shows 3 striped green squiggles, and the third shows 1 striped purple diamond form a set. They show 2, 3, and 1 symbols (each has a different number); they show ovals, squiggles, and diamonds (each shows a different shape); they use colors red, green, and purple (3 different colors); and lastly, they all share the same fill style: striped.

Write a program that finds all sets for 12 provided cards!

Input

The input to your program will consist of 4 lines, each containing 3 strings representing 3 cards, each is of the form `ABCD` where

- `A` is `{1, 2, 3}`, corresponding to the number of symbols.
- `B` is `{D, S, O}`, corresponding to diamonds (D), squiggles (S), and ovals (O).
- `C` is `{S, T, O}`, corresponding to solid (S), striped (T), and open (O) fill styles.
- `D` is `{R, G, P}`, corresponding to red (R), green (G), and purple (P).

Think of the cards as being arranged in the input as follows:

```
+----------+
|  1  2  3 |
|  4  5  6 |
|  7  8  9 |
| 10 11 12 |
+----------+
```

Output

Output all sets you can find, one per line. For each set, output the numbers of the card in the set in sorted order. The sets should be listed in sorted order using the number of their first card, breaking ties using the numbers of the second and third card in the set.

If no sets can be formed, output "`no sets`". (Do not include any punctuation.)

The sample input/output corresponds to the illustration.

Sample Input 1	Sample Output 1
3DTG 3DOP 2DSG 1SOP 1DTG 2OTR 3DOR 3STG 2DSP 3SSP 3OTG 1DTP	1 8 11 2 9 12 3 7 12 5 7 9 6 8 12 7 10 11

Problem D
Cracking The Safe
Time limit: 3 seconds

Your little sister misplaced the code for her toy safe - can you help her?

This particular safe has 9 buttons with digital displays. Each button shows a single digit in the range `0..3`. When you push one of the buttons, the number it displays is incremented by 1, circling around from 3 to 0. However, pushing a button will also increment the other digits in the same row and the same column as the button pushed.

The safe opens when the display shows nine zeros.

For instance, if you pushed the top-left, center, center, and middle-right buttons, in this order, the safe's display would change like so:

```
3 1 2       0 2 3       0 3 3       0 0 3       0 0 0
0 1 1  ->   1 1 1  ->   2 2 2  ->   3 3 3  ->   0 0 0
3 2 3       0 2 3       0 3 3       0 0 3       0 0 0
```

Write a program to determine if the safe can be opened, and if so, how many button pushes it would take!

Input

The input is a single test case, given as 9 digits d, $(0 \leq d \leq 3)$ on 3 lines, representing the digits that are initially displayed on the safe's buttons. Your program will be run multiple times on different inputs.

Output

Output the number of times buttons need to be pushed to open the safe! (The same button may need to be pushed more than once, and you do not have to output which buttons must be pushed.) If the safe cannot be opened, output `-1`.

Sample Input 1	Sample Output 1
3 1 2 0 1 1 3 2 3	4

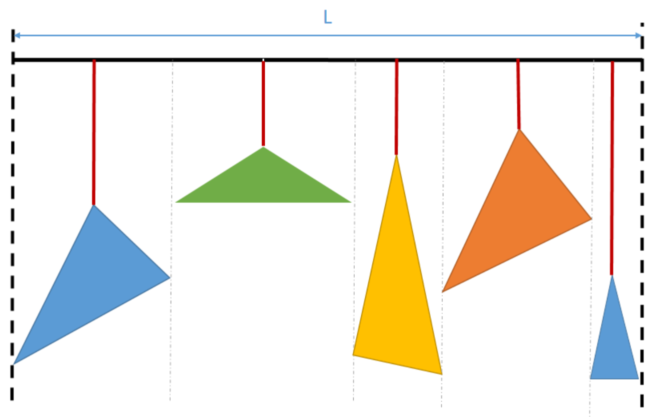
Sample Input 2	Sample Output 2
0 0 3 2 2 3 2 2 1	-1

This page is intentionally left blank.

Problem E
Triangle Ornaments
Time limit: 1 second

A company makes triangle-shaped ornaments for the upcoming holidays. Each ornament is tied at one of its corners to a rod using a string of unknown length. Multiple of these ornaments may be attached to the same rod. These ornaments should be able to swing (rotate around the axis formed by the string) without interfering with each other.

Write a program that computes the minimum required length for the rod, given a list of triangles!



Input

The input consists of a single test case. The first line contains one integer N ($0 < N \leq 100$), denoting the number of triangles. The next N lines each contain three integers A, B, C denoting the lengths of the three sides of each triangle. The triangle will hang from the corner between sides A and B. You are guaranteed that A, B, C form a triangle that has an area that is strictly greater than zero.

Output

Output the required length L such that all triangles can be hung from the rod, no matter how long or short each triangle's string is. No triangle should swing beyond the rod's ends. You may ignore the thickness of each ornament, the width of the string and you may assume that the string is attached exactly to the triangle's end point.

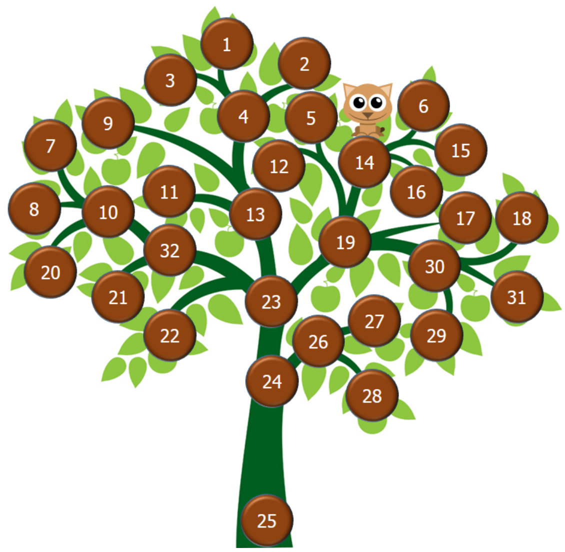
Your answer should be accurate to within an absolute or relative error of 10^{-4}.

Sample Input 1	Sample Output 1
2 3 3 3 4 4 5	8.0

Sample Input 2	Sample Output 2
2 3 3 3 4 5 4	6.843530573929037

Sample Input 3	Sample Output 3
3 7 20 14 7 14 20 14 20 7	20.721166413503266

Problem F
Kitten on a Tree
Time limit: 1 second



Ouch! A kitten got stuck on a tree. Fortunately, the tree's branches are numbered. Given a description of a tree and the position of the kitten, can you write a program to help the kitten down?

Input

The input is a description of a single tree. The first line contains an integer K, denoting the branch on which the kitten got stuck. The next lines each contain two or more integers $a, b_1, b_2, \ldots$. Each such line denotes a branching: the kitten can reach a from $b_1, b_2, \ldots$ on its way down. Thus, a will be closer to the root than any of the b_i. The description ends with a line containing `-1`. Each branch b_i will appear on exactly one line. All branch numbers are in the range $1..100$, though not necessarily contiguous. You are guaranteed that there is a path from every listed branch to the root. The kitten will sit on a branch that has a number that is different than the root.

The illustration above corresponds to the sample input.

Output

Output the path to the ground, starting with the branch on which the kitten sits.

Sample Input 1

```
14
25 24
4 3 1 2
13 9 4 11
10 20 8 7
32 10 21
23 13 19 32 22
19 12 5 14 17 30
14 6 15 16
30 18 31 29
24 23 26
26 27 28
-1
```

Sample Output 1

```
14 19 23 24 25
```

This page is intentionally left blank.

Problem G
First Orchard
Time limit: 6 seconds

Source: Media-mania.de

First Orchard is a cooperative game for children 2 years and up. In this simple game, the players pick colored fruits from trees in an orchard and place them into a basket. To make the game more interesting, a raven tries to reach the orchard to steal the fruits. If the players are successful in moving all fruits into the basket before the raven can get to the orchard, they'll win. If the raven gets there first, the players lose!

Your task is to determine the probability with which the players will win!

The game is played as follows. There are 4 trees in the orchard, each bearing a different type of fruit: red apples, green apples, blue plums, and yellow pears. The raven tries to reach the orchard through a pathway containing one or more steps.

At each turn, the players roll a six-sided die. Four of the faces show a color (red, green, yellow, blue), the fifth face shows a fruit basket, and the sixth face shows a raven. All six faces can appear with equal probability.

- *Red*, *Green*, *Yellow*, *Blue*. In these cases, a fruit with the corresponding color is placed into the basket, provided the tree corresponding to this color still has any fruits left to pick. Otherwise, the players move on to the next turn.
- *Fruit Basket*. If the players roll the 'basket' face, they will pick a fruit from the tree that has the largest number of fruits left. If there are multiple such trees with the same number of fruits, any of the trees can be chosen.
- *Raven*. If the 'raven' face appears on the die, then the raven moves one step closer towards the orchard.

The game is over if either the players have picked all fruits, or if the raven reaches the orchard, whichever happens first. If the raven reaches the orchard before the players have placed all fruits into the basket, the players lose.

Input

The input consists of a single test case with a single line of input. The input contains five integer numbers: `R G B Y S`. R, G, B, Y denote the number of red, green, blue, and yellow fruits initially on each tree, respectively, S denotes the number of steps on the raven's pathway. Since the game is intended for young children, there will be at most 4 fruits of each color: $0 \leq R, G, B, Y \leq 4$. There will be at least one fruit overall: $0 < R+G+B+Y$. The raven will require at least 1 and at most 8 steps: $1 \leq S \leq 8$. The number of steps is equal to the number of times the raven must appear on the die for the players to lose.

Output

Output the probability that the players will win as a floating point number. The absolute error of your result should be less than 10^{-5}.

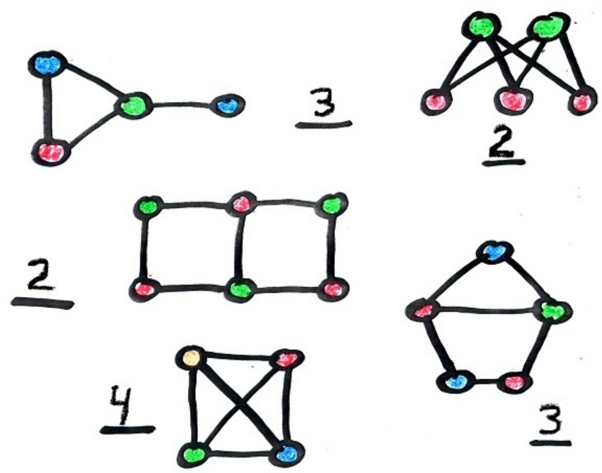
Sample Input 1	Sample Output 1
`1 1 0 0 3`	`0.920138888889`

Sample Input 2	Sample Output 2
`4 4 4 4 5`	`0.631357306601`

Sample Input 3	Sample Output 3
`4 4 4 4 4`	`0.459393713591`

Problem H
Coloring Graphs
Time limit: 6 seconds

To address the impending STEM shortage early on, your local elementary school decided to teach graph theory to its kindergarten students! To tap into their age-specific skills, the students are asked to color the vertices of a graph with colors of their own choosing. There is one constraint, however: they cannot use the same color for two vertices if those vertices are connected by an edge. Furthermore, they are asked to use as few different colors as possible. The illustration shows a few examples of student work.



There is one problem, as you can imagine: there is no money to train teachers to grade these students' submissions! Thus, your task is to write a program that computes the sample solutions for the graphs given on each work sheet!

Input

The input consists of a description of a single graph. The first line contains a number N ($2 \leq N \leq 11$), the number of vertices in the graph. Vertices are numbered $0 \ldots N-1$. The following N lines contain one or more numbers each. The i^{th} line contains a list of vertex numbers v_j, denoting edges from v_i to each v_j in the list. You may assume that the graph is connected (there is a path between any two pairs of vertices).

Output

Output the minimum number of colors required to color all vertices of the graph such that no vertices that share an edge are colored using the same color!

The sample input corresponds to the graphs shown on the illustration.

Sample Input 1	Sample Output 1
4 1 2 0 2 3 0 1 1	3

Sample Input 2	Sample Output 2
5 2 3 4 2 3 4 0 1 0 1 0 1	2

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

Sample Input 4	Sample Output 4
4 1 2 3 0 2 3 0 1 3 0 1 2	4

Sample Input 5	Sample Output 5
5 1 2 0 2 3 0 1 4 1 4 2 3	3