Problem A. Bonsai

Description

After being assaulted in the parking lot by Mr. Miyagi following the “All Valley Karate Tournament", John Kreese has come to you for assistance. Help John in his quest for justice by chopping off all the leaves from Mr. Miyagi's bonsai tree!

You are given an undirected tree (i.e., a connected graph with no cycles), where each edge (i.e., branch) has a nonnegative weight (i.e., thickness). One vertex of the tree has been designated the root of the tree. The remaining vertices of the tree each have unique paths to the root; non-root vertices which are not the successors of any other vertex on a path to the root are known as leaves.

Determine the minimum weight set of edges that must be removed so that none of the leaves in the original tree are connected by some path to the root.

Input

The input will contain multiple test cases. Each test case will begin with a line containing a pair of integers n (where $1 \leq n \leq 1000$) and r (where $r \in \{1, \ldots, n]$) indicating the number of vertices in the tree and the index of the root vertex, respectively. The next $n-1$ lines each contain three integers u_i v_i w_i (where $u_i, v_i \in \{1, \ldots, n\}$ and $0 \leq w_i \leq 1000$) indicating that vertex u_i is connected to vertex v_i by an undirected edge with weight w_i. The input will not contain duplicate edges. The end-of-input is denoted by a single line containing “0 0”.

Output

For each input test case, print a single integer indicating the minimum total weight of edges that must be deleted in order to ensure that there exists no path from one of the original leaves to the root.

Sample Input

```
15 15
1 2 1
2 3 2
2 5 3
5 6 7
4 6 5
6 7 4
5 15 6
15 10 11
10 13 5
13 14 4
12 13 3
9 10 8
```

```
8 9 2
9 11 3
0 0
```

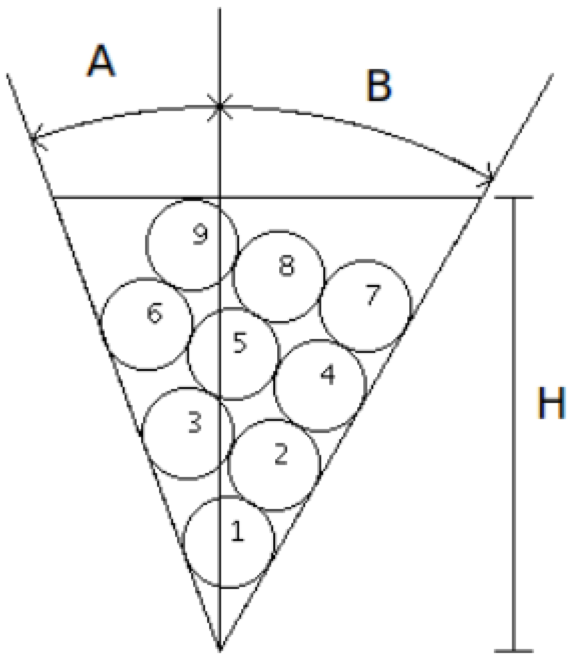
Sample Output

```
16
```

Problem B. Fruit Bowl

Description

For her next project, Beebe Flat wishes to paint a bowl of fruit. Naturally, the bowl is two-dimensional, as is all of the fruit. She bought a triangular bowl which somehow always manages to stay upright, with a perfectly level top. The bowl has a height H, and opens A degrees to the left and B degrees to the right (see diagram). In the interest of art, Beebe eschews symmetry, so $A \neq B$. She plans to buy perfectly circular fruit, each with radius 1, to put into the bowl. However, perfectly circular fruit is expensive, so she needs your help to figure out how much fruit she has to buy. She plans to fill the bowl to the brim, adding as much fruit as possible without any part exceeding the height of the bowl. Thankfully, she is not interested in an optimal packing, since she wants a simple algorithm for actually arranging the fruit. She will place the fruit one at a time, each time choosing the lowest possible location for the center.



Since Beebe doesn't like to deal with ties, she always purchases a bowl such that there is only one lowest possible location within a margin of 10^{-5} . Finally, to ensure that a lid will fit nicely on the bowl when she's done painting, she only chooses bowls such that, when properly packed, the last piece of fruit to fit will be at least 10^{-2} below the top, and the next piece of fruit that would fit if the bowl were taller will jut out at least 10^{-2} above the top.

Given a particular bowl, how many pieces of fruit does Beebe need to buy to fill the bowl in this manner?

Input

The input consists of multiple test cases. Each test case has three integers on a single line, denoting A, B, and H. $1 \leq A, B \leq 45$, $A \neq B$, and $1 \leq H \leq 300$. The last test case will be followed by $A = B = H = 0$, which should not be processed.

Output

For each test case, print on a single line the number of pieces of fruit that Beebe needs to buy to fill the given bowl.

Sample Input

```
20 30 10
10 20 20
0 0 0
```

Sample Output

```
9
25
```

Problem C. Number Game

Description

Carl and Ellie are in the midst of another adventure; this time, it is a road trip through Canada! They've just arrived in Saskatchewan, right in the middle of the Canadian prairies, and, to their horror, have discovered that all the rumors about it being dreadfully flat are true. Suddenly, the warnings their Canadian friends gave them prior to the trip spring back to mind, such as:

It's so flat, a boy can watch his dog run away!

Or:

It's so flat, it's impossible to jump to your death!

As the driver, Carl is worried about falling asleep at the wheel and has decided to come up with a game to relieve the boredom. As a conscientious driver, he doesn't want the game to be too distracting—the roads here are unerringly straight, so it's easy to lose track of motion—so the rules are simple:

* Carl picks a number, N, between 1 and 1,000,000,000.

* Carl and Ellie take turns subtracting an integer (between 1 and 20) from N. Carl plays first, and the winner is the one who subtracts off a number to get 0.

For example, suppose Carl picks the number 50. He subtracts off the number 5, leaving 45. Ellie subtracts off 17, leaving 28. Carl subtracts off 8, leaving 20. Finally, Ellie subtracts off 20, leaving 0, and wins!

Frankly, Ellie would rather sleep than play this game, so she has reprogrammed the GPS (which isn't necessary in this region, anyway) to play for her instead. Her method of choosing a number is straightforward:

* If on a given turn, the number remaining is 20 or less, then she picks that number and wins.

* Otherwise, her choice of number is completely determined by the number Carl just picked, as follows. Before the game starts, Ellie chooses 20 random numbers, $1 \leq a_1, a_2, a_3, \ldots, a_{20} \leq 20$. Then whenever Carl subtracts off the number k, Ellie responds by subtracting off the number a_k (unless she can win).

Carl needs your help! To help stay motivated playing this game, he would like to know if there exists a winning strategy for him, given the numbers N and $a_1, a_2, a_3, \ldots, a_{20}$.

Input

Each input case begins with the number N on a line by itself; the next line contains the numbers $a_1, a_2, \ldots, a_{20}$, separated by spaces. Input terminates with a line containing 0.

Output

For each test case, print “Carl can win” or “Carl can’t win”.

Sample Input

```
42
20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
0
```

Sample Output

```
Carl can’t win
```

In this example, since Carl chooses N to be 42, it turns out that he has no winning strategy. (Indeed, whichever number he initially subtracts off, Ellie will subtract off a number leaving 21 for Carl. Then whatever number Carl subtracts off next, Ellie will be left with a number between 1 and 20, and she wins!)

Problem D. Leap Frog

Description

Jack and Jill play a game called "Leap Frog" in which they alternate turns jumping over each other. Both Jack and Jill can jump a maximum horizontal distance of 10 units in any single jump. You are given a list of valid positions $x_1, x_2, \ldots, x_n$ where Jack or Jill may stand. Jill initially starts at position x_1, Jack initially starts at position x2, and their goal is to reach position x_n.

Determine the minimum number of jumps needed until either Jack or Jill reaches the goal. The two players are never allowed to stand at the same position at the same time, and for each jump, the player in the rear must hop over the player in the front.

Input

The input will contain multiple test cases. Each test case will begin with a single line containing a single integer n (where $2 \leq n \leq 100000$). The next line will contain a list of integers $x_1\ x_2 \ldots x_n$ where $0 \leq x_1 < x_2 \ldots < x_n \leq 1000000$. The end-of-input is denoted by a single line containing "0".

Output

For each input test case, print the minimum total number of jumps needed for both players such that either Jack or Jill reaches the destination, or -1 if neither can reach the destination.

Sample Input

```
6
3 5 9 12 15 17
6
3 5 9 12 30 40
```

Sample Output

```
3
-1
```

Problem E. Universal Oracle

Description

In computer science, an oracle is something that gives you the answer to a particular question. For this problem, you need to write an oracle that gives the answer to everything. But it's not as bad as it sounds; you know that 42 is the answer to life, the universe, and everything.

Input

The input consists of a single line of text with at most 1000 characters. This text will contain only well-formed English sentences. The only characters that will be found in the text are uppercase and lowercase letters, spaces, hyphens, apostrophes, commas, semicolons, periods, and question marks. Furthermore, each sentence begins with a single uppercase letter and ends with either a period or a question mark. Besides these locations, no other uppercase letters, periods, or question marks will appear in the sentence. Finally, every question (that is, a sentence that ends with a question mark) will begin with the phrase "What is..."

Output

For each question, print the answer, which replaces the "What" at the beginning with "Forty-two" and the question mark at the end with a period. Each answer should reside on its own line.

Sample Input

```
Let me ask you two questions. What is the answer to life? What is
the answer to the universe?
```

Sample Output

```
Forty-two is the answer to life.
Forty-two is the answer to the universe.
```

Problem F. Rectangles

Description

A rectangle in the Cartesian plane is specified by a pair of coordinates (x_1, y_1) and (x_2, y_2) indicating its lower-left and upper-right corners, respectively (where $x_1 \le x_2$ and $y_1 \le y_2$). Given a pair of rectangles, $A = ((x_1^A, y_1^A), (x_2^A, y_2^A))$ and $B = ((x_1^B, y_1^B), (x_2^B, y_2^B))$, we write $A \preccurlyeq B$ (i.e. A "precedes" B), if $x_2^A < x_1^A$ and $y_2^A < y_1^B$.

In this problem, you are given a collection of rectangles located in the two-dimension Euclidean plane. Find the length L of the longest sequence of rectangles $(A_1, A_2, \dots, A_L)$ from this collection such that

$$A1 \preccurlyeq A2 \preccurlyeq \dots \preccurlyeq AL.$$

Input

The input will contain multiple test cases. Each test case will begin with a line containing a single integer n (where $1 \le n \le 1000$), indicating the number of input rectangles. The next n lines each contain four integers $x_1^i\ y_1^i\ x_2^i\ y_2^i$ where $-1000000 \le x_1^i \le x_2^i \le 1000000$, $-1000000 \le y_1^i \le y_2^i \le 1000000$, and $1 \le i \le n$), indicating the lower left and upper right corners of a rectangle. The end-of-input is denoted by a single line containing the integer 0.

Output

For each input test case, print a single integer indicating the length of the longest chain.

Sample Input

```
3
1 5 2 8
3 -1 5 4
10 10 20 20
2
2 1 4 5
6 5 8 10
0
```

Sample Output

```
2
1
```

Problem G. Rectangles Too

Note: This problem is almost identical to the previous problem. The single difference between the two problems has been marked below.

Description

A rectangle in the Cartesian plane is specified by a pair of coordinates (x_1, y_1) and (x_2, y_2) indicating its lower-left and upper-right corners, respectively (where $x_1 \leq x_2$ and $y_1 \leq y_2$). Given a pair of rectangles, $A = ((x_1^A, y_1^A), (x_2^A, y_2^A))$ and $B = ((x_1^B, y_1^B), (x_2^B, y_2^B))$, we write $A \preccurlyeq B$ (i.e. A "precedes" B), if $x_2^A < x_1^A$ and $y_2^A < y_1^B$.

In this problem, you are given a collection of rectangles located in the two-dimension Euclidean plane. Find the length L of the longest sequence of rectangles $(A_1, A_2, \ldots, A_L)$ from this collection such that

$$A1 \preccurlyeq A2 \preccurlyeq \ldots \preccurlyeq AL.$$

Input

The input file will contain multiple test cases. Each test case will begin with a line containing a single integer n (where $\underline{\mathbf{1 \leq n \leq 100000}}$), indicating the number of input rectangles. The next n lines each contain four integers $x_1^i\ y_1^i\ x_2^i\ y_2^i$ where $-1000000 \leq x_1^i \leq x_2^i \leq 1000000$, $-1000000 \leq y_1^i \leq y_2^i \leq 1000000$, and $1 \leq i \leq n$), indicating the lower left and upper right corners of a rectangle. The end-of-input is denoted by a single line containing the integer 0.

Output

For each input test case, print a single integer indicating the length of the longest chain.

Sample Input

```
3
1 5 2 8
3 -1 5 4
10 10 20 20
2
2 1 4 5
6 5 8 10
0
```

Sample Output

```
2
1
```

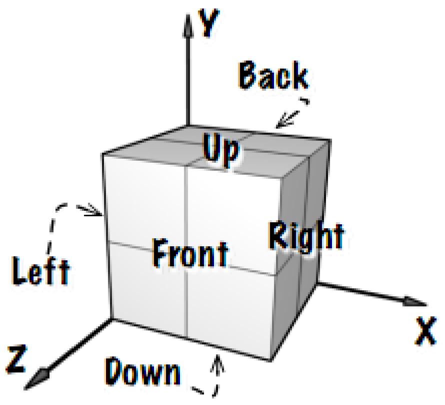
Problem H. Rubik 23

Description

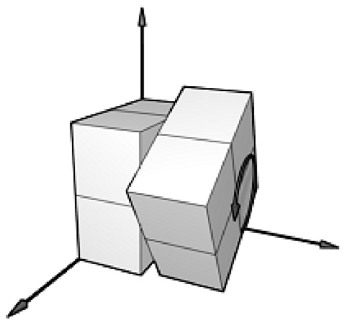
Sonny is probably the only computer science Ph.D. student who cannot solve a Rubik's cube. One day, he came across a neat little 2 × 2 × 2 Rubik's cube, and thought, "Finally, here's a cube that's easy enough for me to do!" Nope, wrong! He got pwned, hardcore. How embarrassing.

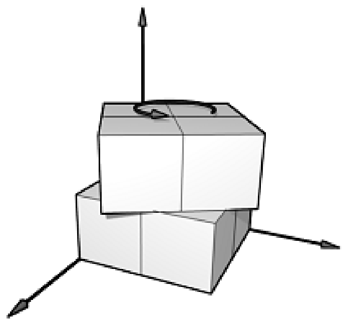
To ensure that this does not happen again, he decides to write a computer program to solve the cube. Then he had this brilliant idea: Why not have the students at the programming contest do the work instead? So, given an initial configuration of the 2 × 2 × 2 Rubik's cube, your task for this problem is to write a program that solves it.

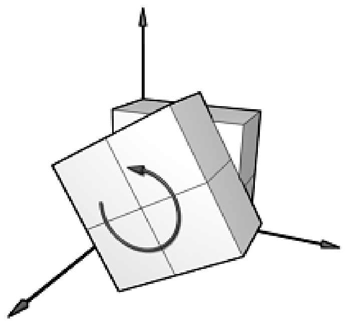
The mini-cube has 6 faces, each with 4 painted tiles on it. The faces are labeled Front (F), Back (B), Up (U), Down (D), Left (L), and Right (R), according to the diagram below. Each of the tiles on the faces can be colored Red (R), Green (G), Blue (B), Yellow (Y), Orange (O), or White (W), and there are exactly 4 instances of each color. The cube is considered solved when the colors of all tiles on each distinct face of the cube match.



You may use any combination of three distinct moves to transform the cube: a turn about the X-axis, a turn about the Y-axis, or a turn about the Z-axis. Each turn is exactly 90 degrees of all tiles on half the cube, in the directions illustrated below. Note that the back-down-left corner is fixed with respect to all valid transforms.


X-axis turn


Y-axis turn


Z-axis turn

Can you come up with a sequence of moves that will solve a given configuration of the Rubik's cube?

Input

You will be given maps of an "unwrapped" cubes showing colors on each of the faces, in the following format:

```
..UU....
..UU....
LLFFRRBB
LLFFRRBB
..DD....
..DD....
```

The letters in the above diagram shows you where to find the colors on each face (as shown in the first diagram) from the map only – it is not valid input! The front face is oriented as in the diagram, with the other faces on the map attached to it so that it wraps to cover the cube. The letters on the faces may be any of R, G, B, Y, O, or W to indicate the color. Dot (.) characters serve to pad the map to a 6 × 8 grid, and are of no other significance.

The input consists of several configuration maps in the format described, separated by blank lines. You may assume that each configuration is both valid and solvable. The end of input is denoted by an "empty" configuration consisting solely of '.' characters; do not process this map.

Output

For each cube, output on a single line a sequence of moves that will solve the cube. Output 'X' for a turn about the X-axis, 'Y' for a turn about the Y-axis, and 'Z' for a turn about the Z-axis. Any sequence of moves (that is reasonably finite) which solves the given configuration will do. (After all, Sonny does need to execute your commands to verify that your program works!) A blank line will suffice for an input cube that is already solved.

Note that the time limit for this problem is more generous than others, and one of the more difficult configurations is provided in the sample.

Sample Input

```
..WO....
..WO....
BBOYGGWR
BBOYGGWR
..YR....
..YR....

..GY....
..BY....
ROYWRRBB
GWOWRBOW
..YG....
..OG....

........
........
........
........
........
........
```

Sample Output

```
X
YZXXXZYZXYXYZZYZZYZXYY
```

Problem I. u Calculate e

Description

A simple mathematical formula for *e* is

$$e = \sum_{i=0}^{n} \frac{1}{i!}$$

where *n* is allowed to go to infinity. This can actually yield very accurate approximations of *e* using relatively small values of *n*.

Output

Output the approximations of e generated by the above formula for the values of n from 0 to 9. The beginning of your output should appear similar to that shown below.

Sample Output

```
n e
- -----------
0 1
1 2
2 2.5
3 2.666666667
4 2.708333333
```