Problem A
In-n-Out
Number of Test Cases: 100
Excution Time Limit: 1 second

The Autumn Festival has started again, and those who stand inside the white zone can participate in the Festival Lottery and win a chance to stand on the podium!

The T-shaped white zone consists of a horizontal strip and a vertical half-strip. As a staff of this event, you're given the coordinates of the strips, and your task is to demonstrate a point inside the T-shaped white zone and another point outside the zone.

Note that, points on the boundary of the strips are considered inside the white zone.

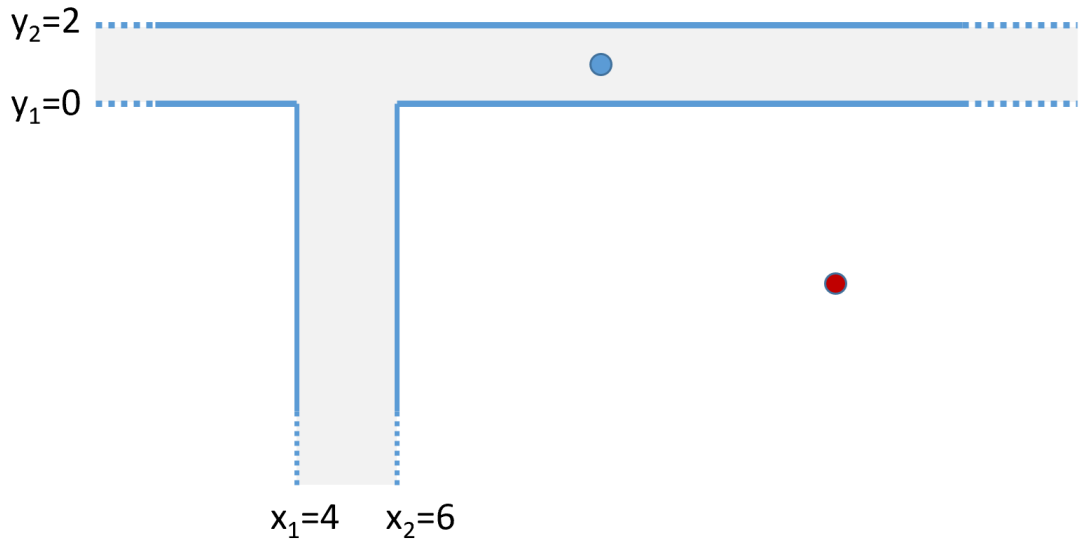

Figure 1: The above figure shows an example with a hozontal strip that spans from $y_1 = 0$ to $y_2 = 2$ and a vertical half-strip that anchors at $y = 0$ and spans from $x_1 = 4$ to $x_2 = 6$. This figures demonstrate two points. The point at $(7, 1)$ is inside the T-shaped white zone, and the point $(8, -6)$ is outside.

Input Format

There will be multiple test cases in the input file, one at a line.

Each test case consists of four integers y_1, y_2, x_1, x_2, which are the y-coordinates of the lower-boundary and the upper-boundary of the horizontal strip and the x-coordinates of the vertical strip, respectively.

A test case with $y_1 = y_2 = x_1 = x_2 = 0$ indicates the end of the input.

You may additionally assume the following.

- $y_1 \leq y_2$, $x_1 \leq x_2$.
- The absolute value of all coordinates does not exceed 10^6.

Output Format

For each test case, print four integers x_i, y_i, x_o, y_o, separated by spaces, where (x_i, y_i) is a point inside the zone and (x_o, y_o) is a point outside the zone.

Your output has to satisfy the following constraint that

- $-10^9 \leq x_i, y_i, x_o, y_o \leq 10^9$.

If there are multiple solutions, print any of them.

Sample Input

```
0 2 4 6
1 1 2 2
0 0 0 0
```

Output for the Sample Input

```
7 1 8 -6
1 2 3 4
```

Problem B
Simplifying Fractions
Number of Test Cases: 10
Excution Time Limit: 1 second

In this problem, you are given a set of fraction numbers expressed as p/q, such as 3/4, 11/5, 20/8, etc. These fraction numbers may not be simplified (reduced) or may be improper (where the numerator is greater than the denominator). Your task is to simplify these fraction numbers and represent any improper fractions as mixed numbers.

Input Format

The input file may contain multiple test cases.

Each test case consists of two positive integers p and q, where p is the numerator and q is the denominator. These two numbers are given in the same line and separated by "/". The values of p and q are not greater than 2^{31}.

The last test case is followed by 0, indicating the end of all test cases.

Output Format

The output should consist of at most three lines:

1. The first line is the numerator.
2. The second line is the integer part and a separating line of many "-"'s. The length of the separating line must be equal to the length of the denominator.
3. The third line is the denominator.

To ensure neat alignment, the numerator and denominator must be right-aligned. If the result is an integer, only one line containing that integer should be printed. If the farction is proper, i.e. $p < q$, print only the fraction part.

Print a "bank line" after every test case, even if it is the last test case.

1 Sample Input

```
5/3
33/99
86/42
5658/123
5806/5754
0
```

Sample Output

```
 2
1-
 3

1
-
3

  1
2--
 21

46

   26
1----
 2877
```

Problem C
Collatz conjecture
Number of Test Cases: 10
Excution Time Limit: 1 second

The Collatz conjecture is one of the most famous unsolved problems in mathematics. The conjecture asks whether repeating two simple arithmetic operations will eventually transform every positive integer into 1. The sequence of integers begins with a positive integer x_0. Each next integer x_{n+1} is obtained from x_n as follows:

$$x_{n+1} = \begin{cases} x_n/2 & \text{if } x_n \text{ is even} \\ 3x_n + 1 & \text{if } x_n \text{ is odd} \end{cases}$$

For example, if $x_0 = 3$, the the sequence of integers are:

$$3, 10, 5, 16, 8, 4, 2, 1$$

Write a program to "test" if Collatz conjecture is true or not. To simplify testing, the number of iterations is limited to 100.

Input Format

The input file may contain multiple test cases.

Each test case contains one positive interger n, $n < 2^{24}$.

The last test case is followed by 0, indicating the end of all test cases.

Output Format

For each test case n, check the Collatz conjecture at most 100 iterations. If $x_i = 1$ for some $0 < i \leq 100$, print "1", otherwise print the value of x_{100}

Sample Input

```
3
31
42
0
```

Sample Output

```
1
10
1
```

Problem D
Maximum Number of Guests at the Party
Number of Test Cases: 5
Excution Time Limit: 1 second

At a party, there are n guests attending, and the arrival and departure times of each guest are recorded. The i-th guest arrives at time a_i and leaves at time d_i. Thus, the i-th guest is at the party during the time interval $[a_i, b_i)$. Write a program to find out the maximum number of guests that are present at the party at the same time.

Input Format

There are multiple test cases in the input file. Each test case is written in several lines and ends with a line containing 0.

Assume guests come and go in groups. Each line of a test case contains 3 integers n, a, and d, means that n guests arrive at the same time a and depart at the same time d. The number of guests n is an integer between 1 and 1000, and the arrival and departure times are integers between 0 and 10000.

The last test case is followed by 0, indicating the end of all test cases.

Output Format

Output the time intervals that has the maximum number of guests present at the party. Each time interval should be maximized, that is, output $[2, 30]$ not $[2, 22][23, 30]$. The output format is as follows: each test case outputs one line, where the first number is the maximum number of guests present at the party at the same time, followed by a list of maximal intervals that have the maximum number of guests.

Sample Input

```
1 1 2
1 0 3
1 2 4
0
1 1 4
1 2 4
1 4 7
1 5 6
```

```
1 3 8
0
3 0 14
2 10 21
2 3 18
3 7 9
1 20 30
3 12 20
2 1 11
0
0
```

Sample Output

```
2: [1,3)
3: [3,4) [5,6)
10: [7,9) [12,14)
```

Problem E
Solving Linear Systems of Equations
Number of Test Cases: 10
Excution Time Limit: 1 second

Write a program to solve a system of linear equations with n unknowns $x_1, x_2, \ldots, x_n$:

$$\begin{aligned} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n &= b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,n}x_n &= b_2 \\ &\vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \cdots + a_{n,n}x_n &= b_n \end{aligned}$$

All coefficients $a_{i,j}$ and b_i, $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, n$, are rational numbers, and the solution should also be expressed in rational numbers. For example: $x_1 = 1/2$ should not be written as $x_1 = 0.5$.

Input Format

The input data consists of multiple test cases.

Each test case starts with an integer n, $n \leq 20$, indicating that there are n unknowns and n equations in this test case.

Following this number n are the n equations. Each of the equation contains $n+1$ rational numbers:
$a_{i,1}, a_{i,2}, \ldots, a_{i,n}, b_i$, $i = 1, 2, \ldots, n$.
These numbers represent the equation:
$a_{i,1}x_1 + a_{i,2}x_1 + \ldots, +a_{i,n}x_1 = b_i$, $i = 1, 2, \ldots, n$.
Every rational number is represented as an integer p or a fraction p/q.

The last test case is followed by 0, indicating the end of all test cases.

Output Format

The output for each system of equations consists of two parts: The first part is the system of equations itself, and the second part is its solution.

If the answer is an integer, it must be printed as an integer; otherwise, it must be printed as a simplified fraction of the from p/q. See the sample output for details.

If the system has no solution, print: "`no solutions`", and if the system has infinitely many solutions, print: "`infinite many solutions`".

Sample Input

```
2
4 -5 8
3 2 12
2
1 2 3
2 4 6
2
1 2 3
1 2 4
2
1/2 -2/3 -1
1/4 1/9 3/2
0
```

Sample Output

```
n = 2
 +4 X1 -5 X2 = 8
 +3 X1 +2 X2 = 12

X1 = 76/23
X2 = 24/23

n = 2
 +1 X1 +2 X2 = 3
 +2 X1 +4 X2 = 6

infinite many solutions

n = 2
 +1 X1 +2 X2 = 3
 +1 X1 +2 X2 = 4

no solutions

n = 2
 +1/2 X1 -2/3 X2 = -1
 +1/4 X1 +1/9 X2 = 3/2

X1 = 4
X2 = 9/2
```

Problem F
Chain code of Objects
Number of Test Cases: 10
Excution Time Limit: 1 second

Consider an object in a digital image whose boundary is specified by a starting point and a sequence of movements. Movement between pixels in an iamge can only be in the 8 direction from the current pixel to its adjacent pixels. The 8 directions are defined as follows.

```
4 3 2
5 * 1
6 7 8
```

A sequence of movements represented by the numbers $1, 2, 3, 4, 5, 6, 7, 8$ are called a *chain code*. In this problem, a chain code is appended by a 0 to indicate the end of the code.

Suppose objects have no holes, but they may not be convex. Typically, objects in a digital image should be specified by a starting point, followed by a chain code. For example, with starting point $(7, 5)$, the object shown at the left of the following figure can be descripbed by a sequence of movements at the right.

```
0000X000
00XXXX00
00XXXXX0
0XXXXXX0        6576767111312313440
0XXXXX00
XXXXX000
XXXX0000
```

In this problem the starting point is not relevant and will be omitted. The chain code representing an object can go clockwise or counterclockwise, but it must be a closed curve describing the boundary of the object. Furthermore, the chain codes used in this problem cannot be self-intersecting, i.e. each pixel appears at most once in the chain code.

Write a program to check the correctness of the chain code of an object in a digital iamge.

Input Format

There may be multiple test cases in the input file.

Each test case contains an object in the digital image. An object begins with an object number m, $m > 0$, and then followed by a chain code $d_1 d_2 \cdots d_n 0$, $0 < n \leq 1000$. Each

object, including its chain code, is stored in a single input line, which may exceed 80 characters.

Note that object numbers can be random or even identical, but they must be positive integers.

The last test case is followed by 0, indicating the end of all test cases.

Output Format

For each test case, find the smallest rectangle that contains the object. The sides of the rectangle must be parallel to the x or y axis, i.e. the object cannot be rotated. Set the coordinates of the lower left corner of the rectangle to $(0, 0)$, and print the objects within the rectangle by printing its bounaries. At each picxel on the boundaries, print a "*". Make sure the x axis goes from left to right and the y axis goes from bottom to top.

After the boundaries of the object is printed, check the validity of the chain code. First, if the chain code is not a closed curve, print "not a closed curve". Then check if the curve is self-intersecting. If the chain code is self-intersecting, print "self-Intersecting". Note that a chain code can be "not a closed curve" and "self-Intersecting". In this case, both error messages should be printed.

Finally, a blank line should be printed to separate the output of each object, even if it is the last object.

Sample Input

```
1 6576767111312313440
2 67871142350
3 44678128224660
4 4467812871142350
0
```

Sample Output

```
Object 1:
    *
  ** *
  *   *
 *   **
 *   *
*  **
****
```

```
Object 2:
 *
*  *
*  *
 **
 ***
not a closed curve

Object 3:
 *   *
* * * *
*  * *
 ** *
self-Intersecting

Object 4:
 *
* *   *
*  *  *
 ** **
    ***
not a closed curve
self-Intersecting
```