Candy Store

You are walking with a friend, when you pass a candy store. You make a comment about how unhealthy their wares are. Your friend issues an interesting challenge: who can be the unhealthiest? Both of you will go into the store with the same amount of money. Whoever buys candy with the most total calories wins!

Since you're a smart computer scientist, and since you have access to the candy store's inventory, you decide not to take any chances. You will write a program to determine the most calories you can buy. The inventory tells you the price and calories of every item. It also tells you that there is so much in stock that you can buy as much of any kind of candy as you want. You can only buy whole pieces of candy.

Input

There will be multiple test cases in the input. Each test case will begin with a line with an integer n ($1 \le n \le 5{,}000$), and an amount of money m ($\$0.01 \le m \le \$100.00$), separated by a single space, where n is the number of different types of candy for sale, and m is the amount of money you have to spend. The monetary amount m will be expressed in dollars with exactly two decimal places, and with no leading zeros unless the amount is less than one dollar. There will be no dollar sign. Each of the next n lines will have an integer c ($1 \le c \le 5{,}000$) and an amount of money p ($\$0.01 \le p \le \$100.00$), separated by a single space, where c is the number of calories in a single piece of candy, and p is the price of a single piece of candy, in dollars and in the same format as m. The input will end with a line containing '**0 0.00**'.

Output

For each test case, output a single integer, indicating the maximum amount of calories you can buy with up to m dollars. Output no spaces, and do not separate answers with blank lines.

Sample Input	Sample Output
2 8.00 700 7.00 199 2.00 3 8.00 700 7.00 299 3.00 499 5.00 0 0.00	796 798

Vive la Difference!

Take any four positive integers: **a**, **b**, **c**, **d**. Form four more, like this:

|a-b| |b-c| |c-d| |d-a|

That is, take the absolute value of the differences of **a** with **b**, **b** with **c**, **c** with **d**, and **d** with **a**. (Note that a zero could crop up, but they'll all still be non-negative.) Then, do it again with these four new numbers. And then again. And again. Eventually, all four integers will be the same. For example, start with `1,3,5,9`:

```
1 3 5 9
2 2 4 8 (1)
0 2 4 6 (2)
2 2 2 6 (3)
0 0 4 4 (4)
0 4 0 4 (5)
4 4 4 4 (6)
```

In this case, the sequence converged in 6 steps. It turns out that in all cases, the sequence converges very quickly. In fact, it can be shown that if all four integers are less than 2^n, then it will take no more than 3*n steps to converge!

Given **a**, **b**, **c** and **d**, figure out just how quickly the sequence converges.

Input

There will be several test cases in the input. Each test case consists of four positive integers on a single line ($1 \le$ **a**,**b**,**c**,**d** $\le$ 2,000,000,000), with single spaces for separation. The input will end with a line with four 0s.

Output

For each test case, output a single integer on its own line, indicating the number of steps until convergence. Output no extra spaces, and do not separate answers with blank lines.

Sample Input

```
1 3 5 9
4 3 2 1
1 1 1 1
0 0 0 0
```

Sample Output

```
6
4
0
```

Do It Wrong, Get It Right

In elementary school, students learn to subtract fractions by first getting a common denominator and then subtracting the numerators. However, sometimes a student will work the problem incorrectly and still arrive at the correct answer. For example, for the problem

$$\frac{5}{4} - \frac{9}{12}$$

one can subtract the numbers in the numerator and then subtract the numbers in the denominator, simplify and get the answer. i.e.

$$\frac{5}{4} - \frac{9}{12} = \frac{-4}{-8} = \frac{4}{8} = \frac{1}{2}$$

For a given fraction ***b***/***n***, your task is to find all of the values ***a*** and ***m***, where ***a***≥**0** and ***m***>**0**, for which

$$\frac{a}{m} - \frac{b}{n} = \frac{a-b}{m-n}$$

Input

There will be several test cases in the input. Each test case will consist of a single line with two integers, ***b*** and ***n*** ($1 \le b, n \le 10^6$) separated by a single space. The input will end with a line with two **0**s.

Output

For each case, output all of the requested fractions on a single line, sorted from smallest to largest. For equivalent fractions, print the one with the smaller numerator first. Output each fraction in the form "***a/m***" with no spaces immediately before or after the "***/***". Output a single space between fractions. Output no extra spaces, and do not separate answers with blank lines.

Sample Input	Sample Output
9 12	0/24 5/20 8/16 8/8 5/4
12 14	0/28 9/21 9/7
4 12	0/24 3/18 3/6
0 0	

A Terribly Grimm Problem

Grimm's conjecture states that to each element of a set of consecutive composite numbers one can assign a distinct prime that divides it.

For example, for the range 242 to 250, one can assign distinct primes as follows:

242	243	244	245	246	247	248	249	250
2	3	61	7	41	13	31	83	5

Given the lower and upper bounds of a sequence of composite numbers, find a distinct prime for each. If there is more than one such assignment, output the one with the smallest first prime. If there is still more than one, output the one with the smallest second prime, and so on.

Input

There will be several test cases in the input. Each test case will consist of a single line with two integers, ***lo*** and ***hi*** ($4 \le lo < hi \le 10^{10}$), separated by a single space. It is guaranteed that all the numbers in the range from ***lo*** to ***hi*** inclusive are composite. The input will end with a line with two **0**s.

Output

For each test case, output the set of unique primes, in order, all on the same line, separated by single spaces. Output no extra spaces, and do not separate answers with blank lines.

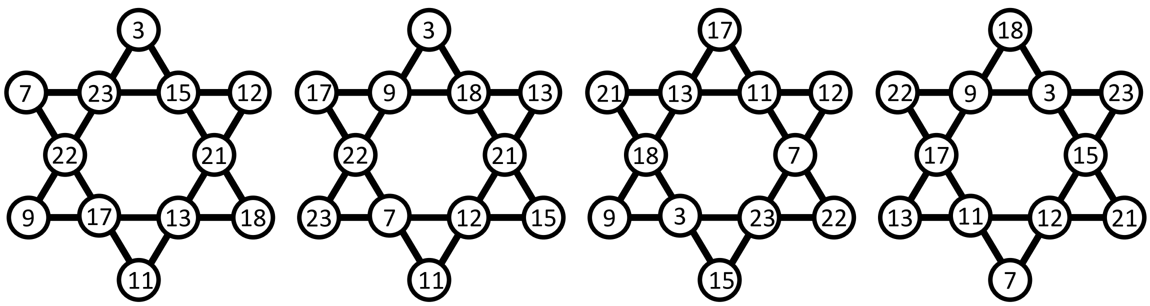
Sample Input	Sample Output
242 250 8 10 0 0	2 3 61 7 41 13 31 83 5 2 3 5

Hexagram

A Hexagram is a 6-pointed star, sometimes called the Star of David. Given these numbers:

```
3 17 15 18 11 22 12 23 21 7 9 13
```

There are four unique ways of assigning the numbers to vertices of the hexagram such that all of the sets of four numbers along the lines have the same sum (57 in this case). All other ways may be obtained from these by rotation and/or reflection.



Given 12 distinct numbers, in how many ways, disregarding rotations and reflections, can you assign the numbers to the vertices such that the sum of the numbers along each of 6 straight lines passing through 4 vertices is the same?

The Input

There will be several test cases in the input. Each test case will consist of twelve unique positive integers on a single line, with single spaces separating them. All of the numbers will be less than 1,000,000. The input will end with a line with twelve 0s.

The Output

For each test case, output the number of ways the numbers can be assigned to vertices such that the sum along each line of the hexagram is the same. Put each answer on its own line. Output no extra spaces, and do not separate answers with blank lines.

Sample Input

```
3 17 15 18 11 22 12 23 21 7 9 13
1 2 3 4 5 6 7 8 9 10 11 13
0 0 0 0 0 0 0 0 0 0 0 0
```

Sample Output

```
4
0
```

Unhappy Numbers

Numbers have feelings too! For any positive integer, take the sum of the squares of each of its digits, and add them together. Take the result, and do it again. A number is Happy if, after repeating this process a finite number of times, the sum is 1. Some happy numbers take more iterations of this process to get to 1 than others, and that would be referred to as its distance from happiness. 1's distance from happiness is 0. 23's distance from happiness is 3, since $2^2 + 3^2 = 13$, $1^2 + 3^2 = 10$, and $1^2 + 0^2 = 1$. Numbers are Unhappy if they are infinitely far away from happiness because they get stuck in a loop.

Given the lower end and upper end of a range of integers, determine how many Unhappy numbers are in that range (inclusive).

Input

There will be several test cases in the input. Each test case will consist of two positive integers, ***lo*** and ***hi*** ($0<lo\leq hi\leq 10^{18}$) on a single line, with a single space between them. Input will terminate with two **0**s.

Output

For each test case, output a single integer on its own line, indicating the count of Unhappy Numbers between ***lo*** and ***hi*** (inclusive). Output no extra spaces, and do not separate answers with blank lines.

Sample Input	Sample Output
1 10 1 100 0 0	7 80

Special Powers

There are certain numbers that think they are special, some are more special than others. These special numbers end with a certain amount of 1's and 2's, and no other digit. For example 1132341221 is special because it ends in 1221, 134242341190 is not a special number. We can classify these special numbers based on how many digits the number ends with that make it special, we will call this the special index of the number. 1132341221 has a special index of 4. One day all of the special numbers had a meeting and started to wonder how many of them are powers of 2, they couldn't find many members there that were, so they thought these special powers must be rare. Your problem is to find these special powers for them.

Input

Each line of input will conist of a single integer $1 \leq n \leq 20$. Input will terminate with a line containing a single 0.

Output

For each line output the smallest non-negative integer k such that 2^k is a special number, with a special index of n.

(for all valid inputs the correct value of k is guaranteed to be less than $2^{63} - 1$)

Sample Input/Output

Input	Output
1	0
2	9
0	