Let's define the function *Suspect*(b, n), where b is an integer that is called the base, and n is an odd integer. It returns one of the boolean values TRUE or FALSE.

- Let t be the highest power of 2 so that 2^t devides $n-1$ and u be the biggest odd integer that devides $n-1$. This means we can write $n-1 = 2^t u$.
- Let x_0 be b^u mod n.
- For all i from 1 to t, let x_i be $(x_{i-1})^2$ mod n.
- If, for any i from 1 to t, $x_i = 1$ and $x_{i-1} <> 1$ and $x_{i-1} <> n-1$, then return FALSE.
- else, if $x_t <> 1$, then return FALSE.
- else return TRUE.

The connaisseur will recognize this function as the essential part of the Miller-Rabin primality test, although it can appear in different forms throughout the literature. In *Cormen et. al.* Ch. 31 this function is called *Witness* and returns the opposite boolean value.

We will call the odd number n suspect in base b if the above function returns TRUE. At the end of this description three examples are given.

It can be proved that whenever the function *Suspect*(b, n) returns FALSE for some base b and an odd number n it is sure that n is not a prime number. The reverse, however, is not the case: whenever *Suspect*(b, n) returns TRUE, there is a high probability that n is a prime number, but we can't be sure. We say that *Suspect*(b, n) fails if it returns TRUE for an n that is not a prime number.

Upto 1000000 there are only 46 failures in base 2, the first three being 2047, 3277 and 4033. In base 3 there are 73 falures, but all of them are different from the base 2 failures, so for every odd number $n < 1000000$ we have that if *Suspect*$(2, n)$ and *Suspect*$(3, n)$ we can be sure that n is a prime number.

Upto 2^{32} we only need to calulate three bases: *Suspect*$(2, n)$, *Suspect*$(7, n)$ and *Suspect*$(61, n)$. If all three function calls return TRUE, it's sure that n is a prime number. This gives us a very quick primality test for all numbers within the range of current day integers.

In this problem we want you to calculate the failures of the function *Suspect*(b, n) in a certain base and for a certain range of numbers n.

Input

The input consists of several lines, each containing three integers: *Base*, *Min* and *Max*. *Base* is an integer between 2 and 1024 (inclusive), *Min* and *Max* will be between 3 and 4294967295 (inclusive). *Max* will not be smaller than *Min*. *Max* will be at most 100000 bigger than *Min*.

A line with three zeroes marks the end of the input; this line should not be processed.

Output

For each line of input, one line of output: "`There are` *NUMBER*1 `odd non-prime numbers between` *NUMBER*2 `and` *NUMBER*3`.`".

If there are odd numbers within this range that fail as suspects in the given base, output an other line: "*NUMBER*4 `suspects fail in base` *NUMBER*5`:`", followed by all failures, in ascending order, each on a line by itself. Use the plural form, even if there is only one failure.

If there are no failures in this range, output the line: "`There are no failures in base` *NUMBER*5.

*NUMBER*1..*NUMBER*5 are to be replaced by the appropriate values.

Separate the cases by a blank line.

Notes:

Suspect$(2, 121)$:

$n-1 = 120 = 8*15 = 2^3*15$, therefore $t = 3, u = 15$
$x_0 = 2^{15} \bmod 121 = 32768 \bmod 121 = 98$
$x_1 = 98^2 \bmod 121 = 9604 \bmod 121 = 45$
$x_2 = 45^2 \bmod 121 = 2025 \bmod 121 = 89$
$x_3 = 89^2 \bmod 121 = 7921 \bmod 121 = 56$

None of the x_i is 1, so the loop test continues until the end. Since x_t is not 1, the function will return FALSE. This is a correct result, since $121 = 11*11$ is composite.

Suspect$(3, 121)$:

$n-1 = 120 = 8*15 = 2^3*15$, therefore $t = 3, u = 15$
$x_0 = 3^{15} \bmod 121 = 14348907 \bmod 121 = 1$
$x_1 = 1^2 \bmod 121 = 1 \bmod 121 = 1$
$x_2 = 1^2 \bmod 121 = 1 \bmod 121 = 1$
$x_3 = 1^2 \bmod 121 = 1 \bmod 121 = 1$

All of the x_i are 1, so the loop test continues until the end. Since x_t is 1, the function will return TRUE. This is a failure!.

Suspect$(3, 89)$:

$n-1 = 88 = 8*11 = 2^3*11$, therefore $t = 3, u = 11$
$x_0 = 3^{11} \bmod 89 = 177147 \bmod 89 = 37$
$x_1 = 37^2 \bmod 89 = 1369 \bmod 89 = 34$
$x_2 = 34^2 \bmod 89 = 1156 \bmod 89 = 88$
$x_3 = 88^2 \bmod 89 = 1 \bmod 89 = 1$

Here we see the loop test again continue to the end. Since x_t is 1, the function will return TRUE. This is correct, because 89 is a prime number.

Sample Input

```
186 800000 900000
2 4000000000 4000001000
3 121 121
0 0 0
```

Sample Output

```
There are 42677 odd non-prime numbers between 800000 and 900000.
4 suspects fail in base 186:
821059
840781
873181
876961

There are 457 odd non-prime numbers between 4000000000 and 4000001000.
There are no failures in base 2.

There are 1 odd non-prime numbers between 121 and 121.
1 suspects fail in base 3:
121
```