Cryptography Reloaded (Act I)

Please click here to download a PDF version of the contest problems. The problem is problem C in the PDF. Remember that you must use stdin/stdout at SPOJ.

What do researchers working at ICPC (Institute for Cryptographic Programming and Computing) do for fun? Well, as you probably have expected, in addition to solving algorithm-related problems on online judges, they also like to toy with various cryptographic schemes. Recently one of the researchers, Tom, has become interested in RSA algorithm implementations used in handheld devices.

Note that the description of the general RSA algorithm is as follows:

1. Choose two distinct prime numbers p and q, and let $n = pq$;
2. Choose an integer e such that $\gcd(e, (p - 1)(q - 1)) = 1$;
3. Compute the integer d that satisfies the congruence relation $de \equiv 1 \pmod{(p - 1) (q - 1)}$.

Then, if person A wants to give person B a way to send an encrypted message to him, A can follow the above steps and release (n, e) as his public key. Upon receiving A's public key, B can simply encrypt message x ($0 \le x < n$) by computing $y = x^e \mod n$. This would result in a message which ideally only A could decrypt with his private key d: $x = y^d \mod n$.

As the computation power of handheld devices is usually limited, a relatively small e is usually used to encrypt data. However this can lead to great security risks. For example, it is quite simple to recover p and q (i.e., factor n) when you have both the public key (n, e) and the private key d. Could you help Tom write a program to demonstrate this?

Input

There are multiple test cases in the input file.

Each test case contains three integers, n, d, and e ($n \le 10^{100}, 3 \le e \le 31$). All three integers are given without any preceding zeros. It is guaranteed that all numbers satisfy the condition as given in the problem statement.

Two successive test cases are separated by a blank line. A case with $n = 0$, $d = 0$ and $e = 0$ indicates the end of the input file, and should not be processed by your program.

For each test case, please print two prime numbers, p and q, such that $n = pq$ and $p < q$, in the format as illustrated below.

Example

Sample Input

```
55 27 3

290203 168101 5

0 0 0
```

Output for the Sample Input

Case #1: 5 11
Case #2: 29 10007