Sonoma State University
Computer Science Department
CS 349 – Spring 2023

Problem:

Jacob Jaffe
Austin Erwin-Maretti

The Problem

The problem consists of finding the modulo of very large integers. Normally, these wouldn't be able to be expressed with the standard library of C++, but using some basic number theory, you'll be able to break the problem into much more manageable parts.

The problem is fairly straightforward… solve for d in the following:

$d = a^b \bmod c$

Normally you could easily solve this by taking std::pow(a, b) % c, but a and b will be 256 bit integers, and c will be a 64 bit integer prime number.

We're often told about the existence modulo arithmetic, and have seldom use for it outside of basic programming problems. It forms the basis for modern cryptography, where the numbers can reach 1024 bit representation. You can see an almost exact implementation of this in RSA cryptography for both encryption and decryption.

Data Structures: You shouldn't need any data structures except a string to deal with the very large integers.

Algorithms: Use dynamic programming with the following equation:
(a * b) mod c = [(a mod c) * (b mod c)] mod c

Think of building it up. If we're looking for 12 mod 3, we could build up by solving
[(4 mod 3) * (3 mod 3)] mod 3.

Input

The input will be formatted as three numbers in lines with spaces between. The first will be a base between 0 and 2^{256} - 1. The second will be a power between 0 and 2^{64} – 1, and the third will be a modulo between 0 and 2^{64} – 1. The last line will be all zeroes.

Example:
102518968917386971168 389276898 15632
0 0 0

Output

The output should be the result of:
(a^b)mod c
a is the first number from the input, b the second number, and c will be the third number.

Example:
123172

If no solution exists, write “no solution”

Test Cases
Input:

```
236 1235 6
12349012675107609 19585 15632
92834609823409862 463984930 89420
12365123461235612346 12341 1641234615
100000000000000000 213587 63462
0 0 0
```

Output:

```
2
1417
37544
766377661
15214
```

Input: