Homework 4

(Time Limit: 1 second)

Problem Description

Given positive integers a, b and c, we want to find the remainder of the division of a^b by c.

Input Format

Each test case contains three positive integers, a, b and c. Two consecutive numbers in a test case are separated by a whitespace character. The input ends with -1, which shall not be processed.

Technical Specifications

There are two subtasks:

Subtask (i) There are at most 6 test cases. Furthermore, $a \leq 100000000$, $b \leq 100000000$ and $c \leq 1000$. Please take care of integer overflows.

Subtask (ii) There are at most 1000 test cases. Furthermore, $a \leq 100000000$, $b \leq 100000000$ and $c \leq 1000$. Because of the time limit of 1 second, efficiency is now an issue. I implemented the method of "recursive doubling" for this subtask.

Output Format

For each test case, output the remainder of the division of a^b by c.

Example

Sample Input:	**Sample Output:**

1 4 2
3 4 5
11 3 121
5 2 7
2 6 3
7 3 10
	1
	1
	0
	4
	1
	3

-1	

Remarks

Because I will test your program using diff, please follow the aforementioned format exactly. For example, please avoid all prompts for inputs.

Solving Subtask (i) and Subtask (ii) earns you 80 and 100 points, respectively.

Recursive Doubling

“Recursive doubling” is efficient for solving this homework. Let me take $b = 90$ for illustration:

- Write 90 in binary so that we know $90 = 64 + 16 + 8 + 2$.
- Calculate $a \bmod c$, $a^2 \bmod c$, $a^4 \bmod c$, $a^8 \bmod c$, $a^{16} \bmod c$, $a^{32} \bmod c$ and $a^{64} \bmod c$ in the following way:

$$\begin{aligned}
a^2 \bmod c &= ((a \bmod c)^2 \bmod c),\\
a^4 \bmod c &= ((a^2 \bmod c)^2 \bmod c),\\
a^8 \bmod c &= ((a^4 \bmod c)^2 \bmod c),\\
a^{16} \bmod c &= ((a^8 \bmod c)^2 \bmod c),\\
a^{32} \bmod c &= ((a^{16} \bmod c)^2 \bmod c),\\
a^{64} \bmod c &= ((a^{32} \bmod c)^2 \bmod c).
\end{aligned}$$

- Having written 90 in binary, we now calculate $a^{90} \bmod c$ as

$$(a^{64} \bmod c) \cdot (a^{16} \bmod c) \cdot (a^8 \bmod c) \cdot (a^2 \bmod c) \bmod c.$$

To avoid integer overflows, you may use long long integers and take your numbers modulo c after every multiplication.