

Area Under Path (areaunderpath)


Peter is standing at the origin of the Cartesian plane, and he decided to take a *regular* walk to point (N, M) for some positive integers N and M . In each step of a *regular* walk, Peter must move parallel to one of the axes. During a move, he can go either one unit to the right or one unit up.


Formally, a regular walk from $(0, 0)$ to (N, M) is a sequence of points (x_i, y_i) ($0 \leq i \leq N + M$) such that

- $(x_0, y_0) = (0, 0)$ and $(x_{N+M}, y_{N+M}) = (N, M)$, and
- for each $i = 1, \dots, N + M$, either $(x_i, y_i) = (x_{i-1} + 1, y_{i-1})$ or $(x_i, y_i) = (x_{i-1}, y_{i-1} + 1)$.

We define the area under a regular walk as the area of the polygon whose vertices in clockwise order are $(0, 0) = (x_0, y_0), (x_1, y_1), \dots, (x_{N+M}, y_{N+M}) = (N, M)$ and $(N, 0)$.

Given a prime number P and a remainder R , you have to find the number W of *regular* walks from $(0, 0)$ to (N, M) under which the area is congruent to R modulo P . Since the answer can be very large, you have to compute it modulo $10^9 + 7$.

 The *modulo* operation ($a \bmod m$) can be written in C/C++/Python as `(a % m)` and in Pascal as `(a mod m)`. To avoid the *integer overflow* error, remember to reduce all partial results through the modulus, and not just the final result!
Notice that if $x < 10^9 + 7$, then $2x$ fits into a C/C++ `int` and Pascal `longint`.

 Among the attachments of this task you may find a template file `areaunderpath.*` with a sample incomplete implementation.

Input

The input file consists of:

- a line containing integers N, M, P, R .

Output






The output file must contain a single line consisting of integer W .

Constraints

- $1 \leq N, M \leq 1\,000\,000$.
- $1 \leq P \leq 100$.
- $0 \leq R < P$.
- P is a prime number.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

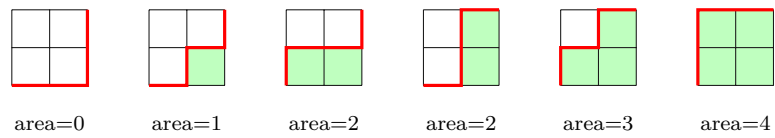
- **Subtask 1** (0 points) Examples.

- **Subtask 2** (16 points) $N, M \leq 10$.

- **Subtask 3** (11 points) $N, M \leq 100$.

- **Subtask 4** (28 points) $N \equiv M \equiv 0 \pmod{P}$.

- **Subtask 5** (45 points) No additional limitations.


Examples

input	output
2 2 3 1	2
2 7 5 3	7

Explanation

In the **first sample case**, there are six possible *regular* walks from $(0,0)$ to (N,M) , as shown in the figure below.



The area under the second and the sixth paths are 1 and 4 respectively, both of which give a remainder of 1 when divided by 3.