CS3230: Programming Assignment 2

Due: Thursday, 21st April 2022, 11:59:59 pm SGT.

This programming assignment is an **optional** component of the module, worth up to 2 bonus marks. Detailed submission instructions can be found at the last page.

Templates are provided for each language. These templates will provide a starting point for your implementation and also provide an implementation of fast input routines. You are recommended to use the templates provided to you. You have to submit your own work, and any form of plagiarism is **not tolerated** and will be subject to disciplinary action.

Task Description

After deciphering the cereal serial code, Csereo has learnt a magic spell to command an army of cereal golems. She starts with $x_1 = 2^m 3^n$ golems, where m and n are non-negative integers.

Smaller golems can combine together to form bigger golems. At the i-th step when Csereo has x_i golems, she can perform the following process:

1. Choose a positive integer x_{i+1} such that x_{i+1} divides x_i.
2. Combine every $d = \frac{x_i}{x_{i+1}}$ golems into one bigger golem. Then Csereo ends up with x_{i+1} golems. To maintain the stability of the spell, the value of d must be strictly larger than x_{i+1}.

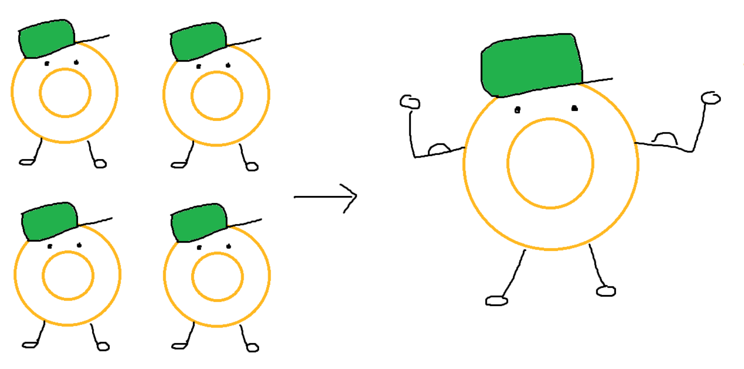

Figure 1: Csereo combines 4 golems into a bigger golem. Here $d = 4$ and $x_{i+1} = 1$.

Eventually, Csereo is satisfied and keeps an army of x_k golems ($k \geq 1$). Csereo wants to find the number of ways to combine her golems, i.e. the number of possible sequences $(x_1, x_2, \ldots, x_k)$ for some k. Can you help her out?

Input

The input will contain two lines. The first line of input will contain a single integer m, while the second line of input will contain the other integer n.

Output

Your program should calculate the number of sequences starting with $x_1 = 2^m 3^n$ that satisfy the given requirements, and output a single integer representing **its remainder modulo $\mathbf{10^9 + 7}$.**

Examples

Example 1	
Input	Output
`3` `0`	`4`

Explanation

$x_1 = 8$. First, Csereo can terminate on x_1 and the resulting sequence is (8). If there exists x_2, then x_2 can only be 1 or 2 since x_2 divides 8 and $x_2 < \sqrt{8}$. $x_2 = 1$ gives only one sequence $(8, 1)$, while $x_2 = 2$ gives two sequences $(8, 2)$ and $(8, 2, 1)$. Therefore, there are 4 sequences starting with 8 satisfying the given requirements.

Example 2	
Input	Output
`2` `1`	`6`

Explanation

$x_1 = 12$. First, Csereo can terminate on x_1 and the resulting sequence is (12). If there exists x_2, then x_2 can only be $3, 2$, or 1 since x_2 divides 12 and $x_2 < \sqrt{12}$. $x_2 = 1$ gives only one sequence $(12, 1)$. $x_2 = 2$ gives two sequences $(12, 2)$ and $(12, 2, 1)$. $x_2 = 3$ gives two other sequences $(12, 3)$ and $(12, 3, 1)$. Therefore, there are 6 sequences starting with 12 satisfying the given requirements.

Example 3	
Input	Output
`5` `5`	`102`

Tasks

Partial Task (1 mark)

To obtain the full 1 mark for the partial task, your algorithm should run in $O(m^2n^2)$ or better.

Full Task (1 mark)

For the full 1 mark, your algorithm should run in $O(mn\min(m,n))$ or better.

Limits

For the partial task, $0 \le m,n \le 100$. For the full task, there are different limits for C++ and Java. For C++, $0 \le m,n \le 500$. For Java, $0 \le m,n \le 400$.

Hints

1. It is guaranteed that $|a\ln 2 + b\ln 3| \ge 10^{-6}$ for integers a and b that are nonzero and have absolute value at most 1000. This means that you can compare magnitudes using floating-point operations since $2^a3^b > 2^c3^d$ if and only if $(a-c)\ln 2 + (b-d)\ln 3 > 0$. You are provided with a method to perform this comparison in the template, which will help you solve the partial task. In Java, using `BigInteger` for comparisons may be too slow.
2. Define $c(m,n)$ to be the number of sequences for 2^m3^n. Calculate $c(m,n)$ using the subproblems $c(i,j)$ where $0 \le i \le m, 0 \le j \le n$, $(i,j) \ne (m,n)$. Derive the recurrence (it helps to observe small cases).

Submission Instructions

You will have to submit in **either Java or C++** to the submission portal at CodeCrunch (`codecrunch.comp.nus.edu.sg`) by **21st April 2022, 23:59**, after which the portal will automatically close. Please check that you have submitted to the **correct language**.

To get the full score for each task, your program needs to solve **ALL** testcases correctly. As this is a programming assignment, not only does your algorithm need to have a good (theoretical) time complexity, but you also need to have an efficient implementation. Your solutions will also be graded using a second set of testcases after the deadline.

Remark

There exists an algorithm to solve the problem in $O(mn\log(mn))$ time.