GPC-UPC First day contest

A. White Christmas Knapsack

1 second, 512 megabytes

In the Black Mirror episode "White Christmas", the main character (Joe) of the story turns out to be in a cabin and the song "I Wish It Could Be Christmas Everyday" is played on the radio. Since Joe had just remembered that he commited a crime related to this song, he tries to destroy the radio. Unfortunately, each time he breaks it, the radio appears again to continue playing the same song.

Now, a burglar has entered to another cabin with similar features, but this time it holds the following: If an object of weight w_i and value v_i is take once, the next time it will appear again with weight w_i and value $\lfloor \frac{v_i}{2} \rfloor$ unless it's new value is 0. He entered with a knapsack that can store objects whose weight sum doesn't exceed C.

Given that the burglar knows that there are n objects in the cabin and their original weights and values, help him know what is the greatest value sum that he can get.

Input

The first line of input contains two integers n and C $\left(1 \leq n \leq 10^3, 1 \leq C \leq 10^3\right)$ — The number of objects and the capacity of the knapsack.

The second line of input contains n integers w_i $\left(1 \leq w_i \leq 10^3\right)$ — the weight of the i-th object.

The third line of input contains n integers v_i $\left(1 \leq v_i \leq 10^6\right)$ — the value of the i-th object.

Output

Print a single line — The answer to the problem.

Scoring

- Subtask 1 (30 points)
  - $1 \leq n \leq 5$
  - $1 \leq v_i \leq 4$
- Subtask 2 (70 points)
  - $1 \leq n \leq 10^3$
  - $1 \leq v_i \leq 10^6$

input
1 41 20 4
output
6

For the sample case we notice that we can take the only object 2 times with values 4 and 2. Unfortunately, we can't take it a third time since it would exceed our knapsack capacity, so we get a final value of 6.

B. Digital Root Power

1 second, 256 megabytes

The digital root of a number is a value obtained by the following recursion:

$$DigitalRoot(n) = \begin{cases} n & n < 10 \\ DigitalRoot(DigitSum(n)) & n \geq 10 \end{cases}$$

Where $DigitSum$ function is self-explanatory.

Given n and k, your task is to compute $(DigitalRoot(n))^k \bmod \left(10^9 + 7\right)$.

Input

The first line of the input contains one integer n $\left(1 \leq n \leq 10^{100000}\right)$ — The base number to get the digital root

The second line of the input contains one integer k $\left(1 \leq k \leq 10^9\right)$ — the power of the digital root to be computed.

Output

Print a single line — The answer to the problem.

Scoring

- Subtask 1 (40 points)
  - $1 \leq n \leq 10^9$
  - $1 \leq k \leq 10^5$
- Subtask 2 (60 points)
  - $1 \leq n \leq 10^{100000}$
  - $1 \leq k \leq 10^9$

input
9 1
output
9

The sample case has $n = 9$, so it's digital root is 9. Thus, our answer is $9^1 \equiv 9 \bmod \left(10^9 + 7\right)$.

C. Study for Exams!

1 second, 256 megabytes

Final exams are coming in $(n + 1)$ days! You need to study hard to pass Algorithms, so you checked the calendar and noticed that there will be some reinforcement lectures for all the students in the previous n days.

Each lecture will be given by a determined professor, so each student attending to the lecture of the i-th professor will improve their knowledge by c_i.

Also, each lecture will have a continuous schedule, $[l_j, r_j], 1 \leq l_j < r_j \leq n$, and if you don't stay the whole lecture, your knowledge won't improve.

You want to improve as much as you can, so you want to attend to some lectures that don't intersect (Two lectures $[l_a, r_a]$, $[l_b, r_b]$ with $l_a \leq l_b$ don't intersect if $r_a \leq l_b$). Try to compute the maximum improvement you can get!

Input

The first line of the input contains three integers n, m and q $\left(1 \leq n, m \leq 10^5, 0 \leq q \leq min\left(\frac{n(n-1)}{2}, 10^5\right)\right)$ — the number of days before the Final exam, the number of professors that can give a lecture and the number of lectures in the schedule.

The second line of the input contains m integers c_i $\left(1 \leq c_i \leq 10^9\right)$ — the knowledge improvement earned by attending to a lecture given by the i-th professor.

The following q lines contain three integers l_i, r_i and k_i ($1 \leq l_i < r_i \leq n, 1 \leq k_i \leq 10^9$) — the limits of the i-th lecture and the id of the professor giving that lecture.

Output

Print a single line — The maximum knowledge improvement you can get from the lectures.

Scoring

- Subtask 1 (10 points)

- $1 \le n \le 6$

- Subtask 2 (40 points)
  - $1 \le n \le 10^3$
- Subtask 3 (50 points)
  - $1 \le n \le 10^5$

input
3 1 2 1000000000 1 2 1 2 3 1
output
2000000000

For the sample case, we have only 1 teacher and 2 lectures given by him. Since the lectures don't intersect we can go to all of them, earning 2000000000 knowledge improvement.

D. Unique Substrings

1 second, 256 megabytes

A *unique* string is a string such that no letter appears twice or more times.

Given a string S, count the number of *unique* substrings.

Input
The first line of input contains a string S $\left(1 \le |S| \le 10^6\right)$ — the given string as stated. The string S consists of only lowercase English letters.

Output
Print a single line — the answer to the problem.

Scoring

- Subtask 1 (100 points)
  - $1 \le n \le 10^6$

input
aba
output
5

For the sample case, we have the following substrings:

- a: unique
- ab: unique
- aba: not unique, letter a repeats
- b: unique
- ba: unique
- a: unique

E. Guess the Number

1 second, 256 megabytes

This is an interactive problem. You have to use a `flush` *operation right after printing each line. For example, in C++ you should use the function* `fflush(stdout)`*, in Java —* `System.out.flush()`*, in Pascal —* `flush(output)` *and in Python —* `sys.stdout.flush()`*.*

In this problem jury has some number x, and you have to guess it. The number x is always an integer from 1 and to n, where n is given to you at the beginning.

You can make queries to the testing system. Each query is a single integer from 1 to n. Flush output stream after printing each query. There are two different responses the testing program can provide:

- the string "<" (without quotes), if the jury's number is less than the integer in your query;
- the string ">=" (without quotes), if the jury's number is greater or equal to the integer in your query.

When your program guessed the number x, print string "! x", where x is the answer, and **terminate your program normally** immediately after flushing the output stream.

Your program is allowed to make no more than 25 queries (not including printing the answer) to the testing system.

Input
Use standard input to read the responses to the queries.

The first line contains an integer n ($1 \le n \le 10^6$) — maximum possible jury's number.

Following lines will contain responses to your queries — strings "<" or ">=". The i-th line is a response to your i-th query. When your program will guess the number print "! x", where x is the answer and terminate your program.

The testing system will allow you to read the response on the query only after your program print the query for the system and perform `flush` operation.

Output
To make the queries your program must use standard output.

Your program must print the queries — integer numbers x_i ($1 \le x_i \le n$), one query per line (do not forget "*end of line*" after each x_i). After printing each line your program must perform operation `flush`.

Each of the values x_i mean the query to the testing system. The response to the query will be given in the input file after you flush output. In case your program guessed the number x, print string "! x", where x — is the answer, and terminate your program.

input
20 > >= >=
output
5 3 4 ! 4