Educational Codeforces Round 60 (Rated for Div. 2)

A. Best Subsegment

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given array $a_1, a_2, \ldots, a_n$. Find the subsegment $a_l, a_{l+1}, \ldots, a_r$ ($1 \le l \le r \le n$) with maximum arithmetic mean $\frac{1}{r-l+1}\sum_{i=l}^{r} a_i$ (in floating-point numbers, i.e. without any rounding).

If there are many such subsegments find the **longest** one.

Input
The first line contains single integer n ($1 \le n \le 10^5$) — length of the array a.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 10^9$) — the array a.

Output
Print the single integer — the length of the longest subsegment with maximum possible arithmetic mean.

Example

input
5 6 1 6 6 0
output
2

Note
The subsegment $[3, 4]$ is the longest among all subsegments with maximum arithmetic mean.

B. Emotes

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

There are n emotes in very popular digital collectible card game (the game is pretty famous so we won't say its name). The i-th emote increases the opponent's happiness by a_i units (we all know that emotes in this game are used to make opponents happy).

You have time to use some emotes only m times. You are allowed to use any emotion once, more than once, or not use it at all. The only restriction is that you **cannot use the same emote more than k times in a row** (otherwise the opponent will think that you're trolling him).

Note that two emotes i and j ($i \ne j$) such that $a_i = a_j$ are considered different.

You have to make your opponent as happy as possible. Find the maximum possible opponent's happiness.

Input
The first line of the input contains three integers n, m and k ($2 \le n \le 2 \cdot 10^5$, $1 \le k \le m \le 2 \cdot 10^9$) — the number of emotes, the number of times you can use emotes and the maximum number of times you may use the same emote in a row.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$), where a_i is value of the happiness of the i-th emote.

Output
Print one integer — the maximum opponent's happiness if you use emotes in a way satisfying the problem statement.

Examples

input
6 9 2 1 3 3 7 4 2
output
54

input
3 1000000000 1 1000000000 987654321 1000000000
output
1000000000000000000

Note
In the first example you may use emotes in the following sequence: $4, 4, 5, 4, 4, 5, 4, 4, 5$.

C. Magic Ship

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You a captain of a ship. Initially you are standing in a point (x_1, y_1) (obviously, all positions in the sea can be described by cartesian plane) and you want to travel to a point (x_2, y_2).

You know the weather forecast — the string s of length n, consisting only of letters U, D, L and R. The letter corresponds to a direction of wind. Moreover, the forecast is periodic, e.g. the first day wind blows to the side s_1, the second day — s_2, the n-th day — s_n and $(n+1)$-th day — s_1 again and so on.

Ship coordinates change the following way:

- if wind blows the direction U, then the ship moves from (x, y) to $(x, y+1)$;
- if wind blows the direction D, then the ship moves from (x, y) to $(x, y-1)$;
- if wind blows the direction L, then the ship moves from (x, y) to $(x-1, y)$;
- if wind blows the direction R, then the ship moves from (x, y) to $(x+1, y)$.

The ship can also either go one of the four directions or stay in place each day. If it goes then it's exactly 1 unit of distance. Transpositions of the ship and the wind add up. If the ship stays in place, then only the direction of wind counts. For example, if wind blows the direction U and the ship moves the direction L, then from point (x, y) it will move to the point $(x-1, y+1)$, and if it goes the direction U, then it will move to the point $(x, y+2)$.

You task is to determine the minimal number of days required for the ship to reach the point (x_2, y_2).

Input
The first line contains two integers x_1, y_1 ($0 \le x_1, y_1 \le 10^9$) — the initial coordinates of the ship.

The second line contains two integers x_2, y_2 ($0 \le x_2, y_2 \le 10^9$) — the coordinates of the destination point.

It is guaranteed that the initial coordinates and destination point coordinates are different.

The third line contains a single integer n ($1 \le n \le 10^5$) — the length of the string s.

The fourth line contains the string s itself, consisting only of letters U, D, L and R.

Output
The only line should contain the minimal number of days required for the ship to reach the point (x_2, y_2).

If it's impossible then print "-1".

Examples

input
0 0 4 6 3 UUU
output
5

input
0 3 0 0 3 UDD
output
3

input
0 0 0 1

1 L

output
-1

Note
In the first example the ship should perform the following sequence of moves: "RRRRU". Then its coordinates will change accordingly: $(0, 0) \to (1, 1) \to (2, 2) \to (3, 3) \to (4, 4) \to (4, 6)$.

In the second example the ship should perform the following sequence of moves: "DD" (the third day it should stay in place). Then its coordinates will change accordingly: $(0, 3) \to (0, 3) \to (0, 1) \to (0, 0)$.

In the third example the ship can never reach the point $(0, 1)$.

D. Magic Gems

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Reziba has many magic gems. Each magic gem can be split into M normal gems. The amount of space each magic (and normal) gem takes is 1 unit. A normal gem cannot be split.

Reziba wants to choose a set of magic gems and split some of them, so the total space occupied by the resulting set of gems is N units. If a magic gem is chosen and split, it takes M units of space (since it is split into M gems); if a magic gem is not split, it takes 1 unit.

How many different configurations of the resulting set of gems can Reziba have, such that the total amount of space taken is N units? Print the answer modulo 1000000007 ($10^9 + 7$). Two configurations are considered different if the number of magic gems Reziba takes to form them differs, or the indices of gems Reziba has to split differ.

Input
The input contains a single line consisting of 2 integers N and M ($1 \leq N \leq 10^{18}$, $2 \leq M \leq 100$).

Output
Print one integer, the total number of configurations of the resulting set of gems, given that the total amount of space taken is N units. Print the answer modulo 1000000007 ($10^9 + 7$).

Examples

input
4 2

output
5

input
3 2

output
3

Note
In the first example each magic gem can split into 2 normal gems, and we know that the total amount of gems are 4.

Let 1 denote a magic gem, and 0 denote a normal gem.

The total configurations you can have is:

- 1111 (None of the gems split);
- 0011 (First magic gem splits into 2 normal gems);
- 1001 (Second magic gem splits into 2 normal gems);
- 1100 (Third magic gem splits into 2 normal gems);
- 0000 (First and second magic gems split into total 4 normal gems).

Hence, answer is 5.

E. Decypher the String

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

This is an interactive problem. Remember to flush your output while communicating with the testing program. You may use `fflush(stdout)` in C++, `system.out.flush()` in Java, `stdout.flush()` in Python or `flush(output)` in Pascal to flush the output. If you use some other programming language, consult its documentation. You may also refer to the guide on interactive problems: https://codeforces.com/blog/entry/45307.

You are given a string t consisting of n lowercase Latin letters. This string was cyphered as follows: initially, the jury had a string s consisting of n lowercase Latin letters. Then they applied a sequence of no more than n (possibly zero) operations. i-th operation is denoted by two integers a_i and b_i ($1 \le a_i, b_i \le n$), and means swapping two elements of the string with indices a_i and b_i. All operations were done in the order they were placed in the sequence. For example, if s is xyz and 2 following operations are performed: $a_1 = 1, b_1 = 2$; $a_2 = 2, b_2 = 3$, then after the first operation the current string is yxz, and after the second operation the current string is yzx, so t is yzx.

You are asked to restore the original string s. Unfortunately, you have no information about the operations used in the algorithm (you don't even know if there were any operations in the sequence). But you may run the same sequence of operations on any string you want, provided that it contains only lowercase Latin letters and its length is n, and get the resulting string after those operations.

Can you guess the original string s asking the testing system to run the sequence of swaps no more than 3 times?

The string s and the sequence of swaps are fixed in each test; the interactor doesn't try to adapt the test to your solution.

Input

Initially the testing system sends one string t, consisting of lowercase Latin letters ($1 \le |t| = n \le 10^4$).

Output

To give the answer, your program should print one line ! s *with a line break in the end*. After that, it should flush the output and terminate gracefully.

Interaction

Before giving the answer, you may submit no more than 3 queries. To ask a query, print one line in the following format: ? s', where s' should be a string consisting of exaclty n lowercase Latin letters. *The line should be ended with a line break character*. After submitting a query, flush the output and read the answer to your query — a string t' consisting of n lowercase Latin letters, which is the result of applying the sequence of swaps to string s'. This string will be given on a separate line ended by a line break character.

If you submit an incorrect query (or ask more than 3 queries), the answer to it will be one string `0`. After receiving such an answer, your program should terminate immediately — otherwise you may receive verdict "Runtime error", "Time limit exceeded" or some other verdict instead of "Wrong answer".

Example

input
yzx aab baa aba
output
? baa ? aba ? aab ! xyz

Note

In the sample, the testcase described in the statement is used. The participant asks the first query with string baa, which is transformed to aab. The second query contains string aba, which is transformed to baa. The third query contains string aab, which is transformed to aba. The participant can deduce that the initial string s was xyz.

Note for hacking phase:

To submit a test in hacking phase, you should provide it in the following format:

The first line should contain the string s you guess, consisting of $n \in [1, 10000]$ lowercase Latin letters.

The second line should contain k ($0 \le k \le n$) — the number of swap operations in the sequence.

Then k lines should follow, i-th of them should denote i-th operation with two integers a_i and b_i ($1 \le a_i, b_i \le n$).

For example, the sample test would look like that:

xyz

2

1 2

2 3

F. Crisp String

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a string of length n. Each character is one of the first p lowercase Latin letters.

You are also given a matrix A with binary values of size $p \times p$. This matrix is symmetric ($A_{ij} = A_{ji}$). $A_{ij} = 1$ means that the string can have the i-th and j-th letters of Latin alphabet adjacent.

Let's call the string *crisp* if **all of the adjacent characters** in it can be adjacent (have 1 in the corresponding cell of matrix A).

You are allowed to do the following move. Choose any letter, remove **all its occurrences** and join the remaining parts of the string without changing their order. For example, removing letter 'a' from "abacaba" will yield "bcb".

The string you are given is *crisp*. The string should remain *crisp* **after every move you make**.

You are allowed to do arbitrary number of moves (possible zero). What is the shortest resulting string you can obtain?

Input

The first line contains two integers n and p ($1 \le n \le 10^5$, $1 \le p \le 17$) — the length of the initial string and the length of the allowed prefix of Latin alphabet.

The second line contains the initial string. It is guaranteed that it contains only first p lowercase Latin letters and that is it *crisp*. Some of these p first Latin letters might not be present in the string.

Each of the next p lines contains p integer numbers — the matrix A ($0 \le A_{ij} \le 1$, $A_{ij} = A_{ji}$). $A_{ij} = 1$ means that the string can have the i-th and j-th letters of Latin alphabet adjacent.

Output

Print a single integer — the length of the shortest string after you make arbitrary number of moves (possible zero).

Examples

input
7 3 abacaba 0 1 1 1 0 0 1 0 0
output
7

input
7 3 abacaba 1 1 1 1 0 0 1 0 0
output
0

input
7 4 bacadab 0 1 1 1 1 0 0 0 1 0 0 0 1 0 0 0
output
5

input
3 3 cbc 0 0 0 0 0 1 0 1 0
output
0

Note

In the first example no letter can be removed from the initial string.

In the second example you can remove letters in order: 'b', 'c', 'a'. The strings on the intermediate steps will be: "abacaba" $\rightarrow$ "aacaa" $\rightarrow$ "aaaa" $\rightarrow$ "".

In the third example you can remove letter 'b' and that's it.

In the fourth example you can remove letters in order 'c', 'b', but not in the order 'b', 'c' because two letters 'c' can't be adjacent.

G. Recursive Queries

time limit per test: 4 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a permutation $p_1, p_2, \ldots, p_n$. You should answer q queries. Each query is a pair (l_i, r_i), and you should calculate $f(l_i, r_i)$.

Let's denote $m_{l,r}$ as the position of the maximum in subsegment $p_l, p_{l+1}, \ldots, p_r$.

Then $f(l, r) = (r - l + 1) + f(l, m_{l,r} - 1) + f(m_{l,r} + 1, r)$ if $l \le r$ or 0 otherwise.

Input

The first line contains two integers n and q ($1 \le n \le 10^6$, $1 \le q \le 10^6$) — the size of the permutation p and the number of queries.

The second line contains n pairwise distinct integers $p_1, p_2, \ldots, p_n$ ($1 \le p_i \le n$, $p_i \ne p_j$ for $i \ne j$) — permutation p.

The third line contains q integers $l_1, l_2, \ldots, l_q$ — the first parts of the queries.

The fourth line contains q integers $r_1, r_2, \ldots, r_q$ — the second parts of the queries.

It's guaranteed that $1 \le l_i \le r_i \le n$ for all queries.

Output

Print q integers — the values $f(l_i, r_i)$ for the corresponding queries.

Example

input
4 5 3 1 4 2 2 1 1 2 1 2 3 4 4 1
output
1 6 8 5 1

Note

Description of the queries:

1. $f(2, 2) = (2 - 2 + 1) + f(2, 1) + f(3, 2) = 1 + 0 + 0 = 1$;
2. $f(1, 3) = (3 - 1 + 1) + f(1, 2) + f(4, 3) = 3 + (2 - 1 + 1) + f(1, 0) + f(2, 2) = 3 + 2 + (2 - 2 + 1) = 6$;
3. $f(1, 4) = (4 - 1 + 1) + f(1, 2) + f(4, 4) = 4 + 3 + 1 = 8$;
4. $f(2, 4) = (4 - 2 + 1) + f(2, 2) + f(4, 4) = 3 + 1 + 1 = 5$;
5. $f(1, 1) = (1 - 1 + 1) + 0 + 0 = 1$.