Problem A. Testing Your Geometry Template

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 megabytes

You are given n distinct points $P_1, P_2, \ldots, P_n$ on the coordinate plane, none of which lie on the x-axis (x-axis is the set of points whose y coordinate is 0).

Let S be the set of all points X on the x-axis such that there exist two integers i, j with $1 \leq i < j \leq n$ such that $XP_i = XP_j$. In other words, S is the set of all points on the x-axis which are equidistant from some two distinct points from $P_1, \ldots, P_n$.

Find the largest distance between any two points in S (not necessarily distinct).

It's guaranteed that in all test cases, S is nonempty and finite.

Input

The first line contains a single integer n ($2 \leq n \leq 200000$) — the number of points.

The i-th of the next n lines contains two integers x_i, y_i ($-10^9 \leq x_i, y_i \leq 10^9$, $y_i \neq 0$) — x and y coordinates of point P_i.

It's guaranteed that S is nonempty and finite in all testcases.

Output

Output a single number — the largest distance between any two points in S (not necessarily distinct).

Your answer is considered correct if its absolute or relative error does not exceed 10^{-6}.

Formally, let your answer be a, and the jury's answer be b. Your answer is accepted if and only if $\frac{|a-b|}{\max(1,|b|)} \leq 10^{-6}$.

Examples

stdin	stdout
3 0 1 -2 5 2 5	14.0000000000

Note

In the sample, $S = \{(-7, 0), (0, 0), (7, 0)\}$.

Note that if S contains a single point, the answer will be 0.

Problem B. Politicians and Competitive Programmers

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

This is an interactive problem

There are n citizens in SmartLand, numbered from 0 to $n-1$. Surprisingly, each of them is either politician or a competitive programmer (but not both), and everyone in the SmartLand knows who the others are. Of course, competitive programmers always tell truth and politicians always lie.

You want to determine, for each citizen of SmartLand, whether he is a politician or a competitive programmer. To do this, you can do the following:

- Choose three **distinct** citizens X, Y, Z, and ask X the following question: "Is it true that Y would answer "Yes" to the question "Is Z competitive programmer?""

However, there is an important constraint: **You can't ask the same citizen twice.** That is, each citizen can be chosen as X at most once.

Determine for each citizen, whether they are competitive programmers or politicians.

Interaction Protocol

Start by reading an integer n ($5 \leq n \leq 1000$) — the number of citizens of SmartLand.

To ask the question, output:

? X Y Z

Here X, Y, Z must be distinct integers with $0 \leq X, Y, Z \leq n - 1$, and X couldn't have been asked questions before.

In response, the interactor will output 1, if X answers "Yes", and 0 otherwise.

When you want to output answer, output:

! s

Here s is a binary string of length n, and s_i should be 1, if citizen i is competitive programmer, and 0 if he is politician.

Make sure to exit immediately to avoid getting other verdicts.

If you ask the invalid query, you will receive a verdict `Wrong Answer`.

After printing a query do not forget to output the end of the line and flush the output. Otherwise, you will get `Idleness limit exceeded` verdict.

To do this, use:

- `fflush(stdout)` or `cout.flush()` in C++;
- `System.out.flush()` in Java;
- `stdout.flush()` in Python;

It is guaranteed that the roles of all citizens are fixed before the start of the interaction. In other words, **the interactor is not adaptive**.

Example

standard input	standard output
5 1 0	? 0 1 2 ? 1 2 3 ! 11110

Note

In the sample, citizens 0, 1, 2, 3 are competitive programmers, and citizen 4 is politician.

Note that the queries in the sample are not sufficient to determine who each citizen is, but nobody forbids you to answer anyways...

Problem C. Fast Squarier Transform

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	256 megabytes

You are given two arrays of integers $[a_1, a_2, \ldots, a_n]$ and $[b_1, b_2, \ldots, b_m]$.

Find the following value:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \lfloor \sqrt{|a_i - b_j|} \rfloor.$$

Note that $\lfloor x \rfloor$ denotes the maximum integer not exceeding x, and $|x|$ denotes the absolute value of x.

Input

The first line of the input contains 2 integers n, m $(1 \leq n, m \leq 10^5)$.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ $(0 \leq a_i,\ a_1 + a_2 + \ldots + a_n \leq 2 \cdot 10^7)$.

The third line of the input contains m integers $b_1, b_2, \ldots, b_m$ $(0 \leq b_i,\ b_1 + b_2 + \ldots + b_m \leq 2 \cdot 10^7)$.

Output

Print a single integer — the sum from the statement.

Examples

standard input	standard output
1 2 1 2 3	2
2 3 1 2 3 4 5	7

Note

In the first sample the answer is $\lfloor \sqrt{|1-2|} \rfloor + \lfloor \sqrt{|1-3|} \rfloor = \lfloor \sqrt{1} \rfloor + \lfloor \sqrt{2} \rfloor = 1 + 1 = 2$.

In the second sample, the answer is $\lfloor \sqrt{|1-3|} \rfloor + \lfloor \sqrt{|1-4|} \rfloor + \lfloor \sqrt{|1-5|} \rfloor + \lfloor \sqrt{|2-3|} \rfloor + \lfloor \sqrt{|2-4|} \rfloor + \lfloor \sqrt{|2-5|} \rfloor = \lfloor \sqrt{2} \rfloor + \lfloor \sqrt{3} \rfloor + \lfloor \sqrt{4} \rfloor + \lfloor \sqrt{1} \rfloor + \lfloor \sqrt{2} \rfloor + \lfloor \sqrt{3} \rfloor = 1 + 1 + 2 + 1 + 1 + 1 = 7$.

Problem D. LIS of Pairs

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

You have n pairs $(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)$ of integers, where $1 \le a_i, b_i \le m$ holds for all i.

Let's define $f(i, j)$ as the length of the longest strictly increasing subsequence of the sequence $\{a_i, b_i, a_j, b_j\}$.

It's clear that $1 \le f(i, j) \le 4$.

For every k from 1 to 4, find the number of ordered pairs (i, j) with $1 \le i, j \le n$, for which $f(i, j) = k$.

Note that a sequence labeled with $\{i_1, i_2, \ldots, i_k\}$ is an increasing subsequence of $\{a_1, a_2, \ldots, a_n\}$ only if:

- $1 \le i_1 < i_2 < \cdots < i_k \le n$

- $a_{i_1} < a_{i_2} < \cdots < a_{i_k}$

Input

The first line of the input contains 2 integers n, m ($1 \le n \le 10^5, 1 \le m \le 10^3$).

The i-th of the following n lines contains 2 integers a_i, b_i ($1 \le a_i, b_i \le m$).

Output

Print 4 integers. The k-th of them should be equal to the number of ordered pairs (i, j) with $1 \le i, j \le n$, for which $f(i, j) = k$.

Examples

standard input	standard output
2 4 1 2 3 4	0 3 0 1
2 1 1 1 1 1	4 0 0 0

Note

In the first sample:

- $f(1, 1) = LIS(1, 2, 1, 2) = 2$.

- $f(1, 2) = LIS(1, 2, 3, 4) = 4$.

- $f(2, 1) = LIS(3, 4, 1, 2) = 2$.

- $f(2, 2) = LIS(3, 4, 3, 4) = 2$.

In the second sample, all numbers are equal, so we wouldn't be able to choose any increasing subsequence of length at least 2.

Problem E. Patriotic Painting 1

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

All points of the real line are initially colored yellow, and you want to make it more colorful. n painters are at your service! If you hire i-th painter, he will paint all the real points in the segment $[L_i, R_i]$ with blue color (that is, for every real x with $L_i \leq x \leq R_i$, if x was colored in blue, it will keep being blue, and if it was yellow, it will become blue).

You like blue color, and initially, you wanted to hire all n painters, but then you decided to save up some money. If some subset of these painters would give the same result, why hire more and pay more?

So, you decided to hire as few painters as possible so that the result of their work would be the same as if you hired all n painters, but there is an issue: you forgot which pair $[L_i, R_i]$ correspondent to which painter.

Find the smallest integer k such that if you hire **any** k distinct painters, the result of their work would be the same as if you hired all n painters.

Here we mean that 2 colorings of the real line are different if there exists a point x which is colored in yellow in one of them and in blue in another.

Input

The first line contains a single integer n ($1 \leq n \leq 200000$) — the number of segments.

The i-th of the next n lines contains 2 integers L, R ($1 \leq L_i < R_i \leq 10^9$). What a pity that you don't remember which of these pairs corresponds to which painter...

Output

Output the smallest integer k such that if you hire any k distinct painters, the end result of their work would be the same as if you hired all n painters. It's easy to see that such k always exists and satisfies $1 \leq k \leq n$.

Examples

standard input	standard output
2 1 42 69 2021	2
3 1 3 1 3 1 3	1
4 1 2 2 3 3 4 1 4	3

Note

In the first sample, there are two painters, one would paint segment $[1, 42]$ in blue, and the other one would paint segment $[69, 2021]$, together they would paint segments $[1, 42]$ and $[69, 2021]$ in blue. It's easy to see that $k = 1$ wouldn't work: if you chose a painter who would paint points in $[1, 42]$, point 69.5 would remain yellow.

In the second sample, there are three painters, and all of them would paint the same segment $[1, 3]$, so if you hired all of them, only segment $[1, 3]$ would be painted! What a waste of resources. $k = 1$ is sufficient, as no matter what painter you hire, he would paint the same segment $[1, 3]$.

In the second sample, there are four painters, one of which would paint segment $[1, 2]$, another $[2, 3]$, another $[3, 4]$, and the last one $[1, 4]$. If you hired all of them, they would paint segment $[1, 4]$ in blue. $k = 1, 2$ are not sufficient: if you hire only painters corresponding to segments $[1, 2]$, $[2, 3]$, only segment $[1, 3]$ would be colored blue in the end, and point 3.5 would remain yellow. It's easy to see that any subset of three painters would color all points from segment $[1, 4]$ in blue, so the answer is 3.

Problem F. Patriotic Painting 2

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

You are given a strip $1 \times n$, consisting of n yellow cells.

You will do k operations with this strip. In the i-th operation, you must select some **exactly** a_i consecutive cells of this strip, and paint all of them blue (if the cell was yellow, it will turn blue, and if it was blue, it will remain blue).

How many different colorings of the strip can you get? Since this number can be very large, print it modulo $10^9 + 7$.

Two colorings of the strip are considered different if some cell is painted yellow in one of them and blue in the other one.

Input

The first line contains two integers n, k ($1 \leq n \leq 10^9, 1 \leq k \leq 4$).

The second line contains k integers $a_1, \ldots, a_k$ ($1 \leq a_i \leq n$).

Output

Output the number of different colorings of the strip that you can get modulo $10^9 + 7$.

Examples

standard input	standard output
10 3 2 2 8	6
5 2 1 2	13
1000000000 4 2020 322 1488 1337	462140823

Problem G. If You Are Homeless... Just Build a House

Input file:	`stdin`
Output file:	`stdout`
Time limit:	4 seconds
Memory limit:	256 megabytes

Your Minecraft world looks like a grid $n \times m$, the cell at the intersection of the i-th row and the j-th column is denoted as (i, j). There are $a_{i,j}$ blocks on the cell (i, j).

With a cost of one coin, you can remove one block from the top of the cell, if there is at least one block there, or add one block on top. In other words, for one coin you can change any $a_{i,j}$ by 1 if it doesn't become negative after.

You want to build a house the size of $h \times w$. To do this, you need a section of the grid of size $h \times w$, in which the numbers of blocks in all cells are the same. In other words, to build a house there have to exist some integers x, y with $1 \le x \le n - h + 1$, $1 \le y \le m - w + 1$ that all numbers $a_{x+i,y+j}$ for $0 \le i < h$, $0 \le j < w$ are equal to each other.

What is the least amount of coins you need to spend to have the necessary area for your new amazing house?

Input

The first line contains two integers n, m $(1 \le n, m \le 500)$ — the size of your Minecraft world.

The second line contains two integers h, w $(1 \le h \le min(n, 20)$, $1 \le w \le min(m, 200))$ — the dimensions of your future house.

The i th of the next n rows contains m integers $a_{i,1}, a_{i,2}, \ldots, a_{i,m}$ $(0 \le a_{i,j} \le 100000)$.

Output

Print a single integer — the least amount of coins you have to spend to have the necessary area for your house.

Examples

stdin	stdout
2 3 1 2 1 2 3 4 5 6	1
2 3 1 2 1 6 3 4 2 5	2

Note

In the first sample, you can increase $a_{1,1}$ with 1 coin, obtaining an area of size 1×2, on each cell of which there are exactly 2 cubes.

In the second example, you can reduce $a_{2,1}$ by 2 with 2 coins, again obtaining an area of size 1×2, on each cell of which there are exactly 2 cubes.

Problem H. Another ICPC String Problem

Input file:	standard input
Output file:	standard output
Time limit:	2 seconds
Memory limit:	512 megabytes

You are given a string s of length n. Process q queries of the following 2 types:

1. Change the i-th character of the string (i.e. s_i) into c;

2. Answer the query: how many times does p occur as a substring of s?

   That is, count the number of indices i such that $1 \leq i \leq n - m + 1$ and $s_i s_{i+1} \ldots, s_{i+m-1} = p_1 p_2 \ldots p_m$.

Find the result of each query of type 2.

Input

The first line of the input contains 2 integers n, q ($1 \leq n \leq 5 \times 10^4, 1 \leq q \leq 10^5$).

The second line of the input contains a string s of length n consisting of lowercase Latin characters.

q lines follow, which may have one of the following formats:

1. If it's a query of the first type, it will contain an integer k_i ($1 \leq k_i \leq n$) and a lowercase character c_i, which means that character s_{k_i} becomes c_i

2. If it's a query of the second type, it will contain a 0 followed by a nonempty string p of lowercase characters, meaning that you should find the number of times that p occurs as a substring of s.

It is guaranteed that there is at least one query of type 2.

It is guaranteed that the sum of lengths of strings p over all queries of the type 2 doesn't exceed $5 \cdot 10^4$.

Output

For each operation of type 2 output the number of times that the query string appears as a substring of s.

Examples

standard input	standard output
5 5 aaaaa 0 aa 2 b 0 aba 4 b 0 aba	4 1 2
1 1 a 0 a	1

Problem I. Vovochka

Input file:	standard input
Output file:	standard output
Time limit:	1 second
Memory limit:	256 megabytes

You are given an array of $3n$ integers $[a_1, a_2, \ldots, a_{3n}]$.

Vovochka divided all the elements of this array into n triplets, calculated the median for each triple, and put the obtained numbers in the sorted order in the array $[b_1, b_2, \ldots, b_n]$.

For example, the array $[1, 2, 3, 1, 2, 3]$ could be split into triplets $(1, 2, 2)$ and $(1, 3, 3)$. In this case, the median array would be $[2, 3]$.

Another way to split into triples is $(1, 2, 3), (1, 2, 3)$, then the array of medians would be $[2, 2]$.

How many distinct arrays could Vovochka get? Since this number can be very large, print it modulo $10^9 + 7$.

As a reminder, the median of three numbers is defined as follows: let $a \le b \le c$ be these three numbers in sorted order, then the median is the number b.

Input

The first line contains a single integer n ($1 \le n \le 2000$).

The second line contains $3n$ integers $a_1, a_2, \ldots, a_{3n}$ ($1 \le a_i \le 3n$) — elements of the array.

Output

Print a single integer — the number of possible arrays $[b_1, b_2, \ldots, b_n]$, modulo $10^9 + 7$.

Examples

standard input	standard output
2 1 2 2 2 2 3	1
2 1 2 3 1 2 3	4
3 9 8 7 6 5 4 3 2 1	21

Note

In the first example, b will always be equal to $[2, 2]$.

In the second example, 4 arrays b are possible: $[1, 2], [1, 3], [2, 2], [2, 3]$.

Problem J. Typical ICPC Problem

Input file:	standard input
Output file:	standard output
Time limit:	6 seconds
Memory limit:	512 megabytes

Suppose that you advanced to the ICPC finals. Of course, there will be some cactus problem! To make all participants suffer, even more, it's an edge cactus. As a cherry on top, the input format is unusual. Perfect problem!

You are given an edge cactus on n nodes. Each node contains an integer from 1 to n such that the numbers in the nodes form a permutation. Let's call a pair of numbers (l, r) with $1 \leq l \leq r \leq n$ **good**, if the following condition holds:

- Consider only those nodes, integers on which are in a range $[l, r]$. Consider only those edges between these nodes, which are present in the original cactus. Then the pair (l, r) is called good if and only if the obtained graph is connected.

Count the number of good pairs.

As a reminder, an edge cactus is a graph such that each edge in it is contained in at most one simple cycle.

As another reminder, a simple cycle is a sequence of distinct nodes (at least 3), such that every two adjacent nodes are connected with an edge, and also the first and the last nodes are connected with an edge.

Input

The first line of the input contains 2 integers $(1 \leq n, m \leq 10^6)$. The edges of the cactus are given by m edge-simple paths.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ $(1 \leq a_i \leq n$, all a_i are **distinct**), where a_i is the number written in the i-th node.

Next m lines contain descriptions of paths, each of which looks as follows:

The first number in a line is an integer k $(1 \leq k \leq 1000)$ — the number of nodes on the path. k integers $v_1, v_2, \ldots, v_k$ $(1 \leq v_i \leq n$, v_i are **not neccessarily distinct**) follow, meaning that there is an edge between nodes v_i and v_{i+1} for each i from 1 to $k-1$. Note that the same node can appear more than once here.

It's guaranteed that each edge will be described at most once and that there are no multi-edges or self-loops.

Output

Output a single integer — the number of pairs (l, r) $(1 \leq l \leq r \leq n)$, such that pair (l, r) is good.

Examples

standard input	standard output
3 2 2 1 3 2 1 2 2 2 3	5
5 2 2 1 3 4 5 4 1 2 3 1 4 3 4 5 3	15

Note

In the first sample we have a cactus with edges $(1, 2)$ and $(2, 3)$. Pairs $(1, 1), (2, 2), (3, 3), (1, 2), (1, 3)$ are good, while pair $(2, 3)$ isn't.

In the second sample we have a cactus with edges $(1, 2)$, $(2, 3)$, $(3, 1)$, $(3, 4)$, $(4, 5)$, $(5, 3)$.

Problem K. ICPC and Typos

Input file:	stdin
Output file:	stdout
Time limit:	1 second
Memory limit:	256 megabytes

Typos are a very important part of ICPC problem statements. ICPC research department found out that there exist 3 main categories of typos:

- One letter of the word was accidentally skipped (for example, `sell` is written instead of `shell`)

- One extra letter was accidentally inserted (for example, `horny` is written instead of `horn`)

- One letter was accidentally replaced with another (for example, `ictc` is written instead of `icpc`)

ICPC research department now considers an alphabet consisting of first n Latin letters and a word s. They wonder: how many distinct words can you obtain by making in s **exactly one** typo?

Input

The first line contains a single integer n ($1 \leq n \leq 26$) — the size of the alphabet.

The second line contains a string s ($2 \leq |s| \leq 100000$), which consists only of the first n letters of the Latin alphabet.

Output

Output the number of distinct words that you can obtain by making in s **exactly one** typo.

Note that the word with typos can contain only letters among the first n letters of the Latin alphabet, too. For example, for $n = 3$ and $s = $ `cab` you can't obtain `dab` by mistyping `c`.

Examples

stdin	stdout
4	32
abcd	

Note

From the string `abcd` from the sample we can obtain the following strings:

```
abc   abd   acd   bcd
aacd  abad  abbd  abca
abcb  abcc  abdd  accd
adcd  bbcd  cbcd  dbcd
aabcd abacd abbcd abcad
abcbd abccd abcda abcdb
abcdc abcdd abdcd acbcd
adbcd babcd cabcd dabcd
```