A. Shooshuns and Sequence

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

One day shooshuns found a sequence of n integers, written on a blackboard. The shooshuns can perform one operation with it, the operation consists of two steps:

1. Find the number that goes k-th in the current sequence and add the same number to the end of the sequence;
2. Delete the first number of the current sequence.

The shooshuns wonder after how many operations all numbers on the board will be the same and whether all numbers will ever be the same.

Input

The first line contains two space-separated integers n and k ($1 \le k \le n \le 10^5$).

The second line contains n space-separated integers: $a_1, a_2, ..., a_n$ ($1 \le a_i \le 10^5$) — the sequence that the shooshuns found.

Output

Print the minimum number of operations, required for all numbers on the blackboard to become the same. If it is impossible to achieve, print -1.

Sample test(s)

input

```
3 2
3 1 1
```

output

```
1
```

input

```
3 1
3 1 1
```

output

```
-1
```

Note

In the first test case after the first operation the blackboard will have sequence `[1, 1, 1]`. So, one operation is enough to make all numbers the same. Thus, the answer equals one.

In the second test case the sequence will never consist of the same numbers. It will always contain at least two distinct numbers 3 and 1. Thus, the answer equals -1.

B. Cosmic Tables

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

The Free Meteor Association (FMA) has got a problem: as meteors are moving, the Universal Cosmic Descriptive Humorous Program (UCDHP) needs to add a special module that would analyze this movement.

UCDHP stores some secret information about meteors as an $n \times m$ table with integers in its cells. The order of meteors in the Universe is changing. That's why the main UCDHP module receives the following queries:

- The query to swap two table rows;
- The query to swap two table columns;
- The query to obtain a secret number in a particular table cell.

As the main UCDHP module is critical, writing the functional of working with the table has been commissioned to you.

Input

The first line contains three space-separated integers n, m and k ($1 \le n, m \le 1000$, $1 \le k \le 500000$) — the number of table columns and rows and the number of queries, correspondingly.

Next n lines contain m space-separated numbers each — the initial state of the table. Each number p in the table is an integer and satisfies the inequality $0 \le p \le 10^6$.

Next k lines contain queries in the format "s_i x_i y_i", where s_i is one of the characters "c", "r" or "g", and x_i, y_i are two integers.

- If s_i = "c", then the current query is the query to swap columns with indexes x_i and y_i ($1 \le x, y \le m, x \ne y$);
- If s_i = "r", then the current query is the query to swap rows with indexes x_i and y_i ($1 \le x, y \le n, x \ne y$);
- If s_i = "g", then the current query is the query to obtain the number that located in the x_i-th row and in the y_i-th column ($1 \le x \le n$, $1 \le y \le m$).

The table rows are considered to be indexed from top to bottom from 1 to n, and the table columns — from left to right from 1 to m.

Output

For each query to obtain a number (s_i = "g") print the required number. Print the answers to the queries in the order of the queries in the input.

Sample test(s)

input

```
3 3 5
1 2 3
4 5 6
7 8 9
g 3 2
r 3 2
c 2 3
g 2 2
g 3 2
```

output

```
8
9
6
```

input

```
2 3 3
1 2 4
3 1 5
c 2 1
r 1 2
g 1 3
```

output

```
5
```

Note

Let's see how the table changes in the second test case.

After the first operation is fulfilled, the table looks like that:

2 1 4

1 3 5

After the second operation is fulfilled, the table looks like that:

1 3 5

2 1 4

So the answer to the third query (the number located in the first row and in the third column) will be 5.

C. Reducing Fractions

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

To confuse the opponents, the Galactic Empire represents fractions in an unusual format. The fractions are represented as two sets of integers. The product of numbers from the first set gives the fraction numerator, the product of numbers from the second set gives the fraction denominator. However, it turned out that the programs that work with fractions in this representations aren't complete, they lack supporting the operation of reducing fractions. Implement this operation and the Empire won't forget you.

Input

The first input line contains two space-separated integers n, m $(1 \le n, m \le 10^5)$ that show how many numbers the first set (the numerator) and the second set (the denominator) contain, correspondingly.

The second line contains n space-separated integers: $a_1, a_2, ..., a_n$ $(1 \le a_i \le 10^7)$ — the numbers that are multiplied to produce the numerator.

The third line contains m space-separated integers: $b_1, b_2, ..., b_m$ $(1 \le b_i \le 10^7)$ — the numbers that are multiplied to produce the denominator.

Output

Print the answer to the problem in the form, similar to the form of the input data. The number of values in the sets you print n_{out}, m_{out} must satisfy the inequality $1 \le n_{out}, m_{out} \le 10^5$, and the actual values in the sets $a_{out, i}$ and $b_{out, i}$ must satisfy the inequality $1 \le a_{out, i}, b_{out, i} \le 10^7$.

Separate the values in the lines by spaces. The printed fraction must be reduced, that is, there mustn't be such integer x $(x > 1)$, that the numerator and the denominator of the printed fraction are divisible by x. If there are several matching answers, print any of them.

Sample test(s)

input
```
3 2
100 5 2
50 10
```
output
```
2 3
2 1
1 1 1
```

input
```
4 3
2 5 10 20
100 1 3
```
output
```
1 1
20
3
```

Note

In the first test sample the numerator equals 1000, the denominator equals 500. If we reduce fraction 1000/500 by the greatest common divisor of the numerator and the denominator (by 500), we obtain fraction 2/1.

In the second test sample the numerator equals 2000, the denominator equals 300. If we reduce fraction 2000/300 by the greatest common divisor of the numerator and the denominator (by 100), we obtain fraction 20/3.

D. Olympiad

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

A boy named Vasya has taken part in an Olympiad. His teacher knows that in total Vasya got at least x points for both tours of the Olympiad. The teacher has the results of the first and the second tour of the Olympiad but the problem is, the results have only points, no names. The teacher has to know Vasya's chances.

Help Vasya's teacher, find two numbers — the best and the worst place Vasya could have won. Note that the total results' table sorts the participants by the sum of points for both tours (the first place has the participant who has got the most points). If two or more participants have got the same number of points, it's up to the jury to assign places to them according to their choice. It is guaranteed that each participant of the Olympiad participated in both tours of the Olympiad.

Input

The first line contains two space-separated integers n, x $(1 \le n \le 10^5; 0 \le x \le 2 \cdot 10^5)$ — the number of Olympiad participants and the minimum number of points Vasya earned.

The second line contains n space-separated integers: $a_1, a_2, ..., a_n$ $(0 \le a_i \le 10^5)$ — the participants' points in the first tour.

The third line contains n space-separated integers: $b_1, b_2, ..., b_n$ $(0 \le b_i \le 10^5)$ — the participants' points in the second tour.

The participants' points are given in the arbitrary order. It is guaranteed that Vasya was present in the Olympiad — there are two integers i, j $(1 \le i, j \le n)$ such, that $a_i + b_j \ge x$.

Output

Print two space-separated integers — the best and the worst place Vasya could have got on the Olympiad.

Sample test(s)

input

```
5 2
1 1 1 1 1
1 1 1 1 1
```

output

```
1 5
```

input

```
6 7
4 3 5 6 4 4
8 6 0 4 3 4
```

output

```
1 5
```

Note

In the first text sample all 5 participants earn 2 points each in any case. Depending on the jury's decision, Vasya can get the first (the best) as well as the last (the worst) fifth place.

In the second test sample in the best case scenario Vasya wins again: he can win 12 points and become the absolute winner if the total results' table looks like that — {4:8, 6:4, 3:6, 4:4, 4:3, 5:0}.

In this table all participants are sorted by decreasing points and we can see how much a participant earned in the first and in the second tour.

In the worst case scenario Vasya can get the fifth place if the table looks like that — {4:8, 4:6, 6:4, 5:4, 4:3, 3:0}, and he earned 4 and 3 points in the first and second tours, correspondingly.

E. Decoding Genome

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

Recently a top secret mission to Mars has taken place. As a result, scientists managed to obtain some information about the Martian DNA. Now we know that any Martian DNA contains at most m different nucleotides, numbered from 1 to m. Special characteristics of the Martian DNA prevent some nucleotide pairs from following consecutively in this chain. For example, if the nucleotide 1 and nucleotide 2 can not follow consecutively in the Martian DNA, then the chain of nucleotides [1, 2] is not a valid chain of Martian DNA, but the chain of nucleotides [2, 1] can be a valid chain (if there is no corresponding restriction). The number of nucleotide pairs that can't follow in the DNA chain consecutively, is k.

The needs of gene research required information about the quantity of correct n-long chains of the Martian DNA. Your task is to write a program that will calculate this value.

Input

The first line contains three space-separated integers n, m, k ($1 \le n \le 10^{15}$, $1 \le m \le 52$, $0 \le k \le m^2$).

Next k lines contain two characters each, without a space between them, representing a forbidden nucleotide pair. The first character represents the first nucleotide in the forbidden pair, the second character represents the second nucleotide.

The nucleotides with assigned numbers from 1 to 26 are represented by English alphabet letters from "`a`" to "`z`" (1 is an "`a`", 2 is a "`b`", ..., 26 is a "`z`"). Nucleotides with assigned numbers from 27 to 52 are represented by English alphabet letters from "`A`" to "`Z`" (27 is an "`A`", 28 is a "`B`", ..., 52 is a "`Z`").

It is guaranteed that each forbidden pair occurs at most once in the input. It is guaranteed that nucleotide's numbers in all forbidden pairs cannot be more than m. Note that order is important in nucleotide pairs.

Please, do not use the `%lld` specifier to read or write 64-bit integers in C++. It is preferred to use `cin`, `cout` streams or the `%I64d` specifier.

Output

Print a single integer — the sought number modulo 1000000007 $(10^9 + 7)$.

Sample test(s)

input
```
3 3 2
ab
ba
```
output
```
17
```

input
```
3 3 0
```
output
```
27
```

input
```
2 1 1
aa
```
output
```
0
```

Note

In the second test case all possible three-nucleotide DNAs are permitted. Each nucleotide can take one of three values, thus in total there are 27 distinct three nucleotide DNAs.

In the third test sample we cannot make any DNA of two nucleotides — the only possible nucleotide "`a`" cannot occur two times consecutively.