A. Sereja and Algorithm

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

Sereja loves all sorts of algorithms. He has recently come up with a new algorithm, which receives a string as an input. Let's represent the input string of the algorithm as $q = q_1q_2... \ q_k$. The algorithm consists of two steps:

1. Find any continuous subsequence (substring) of three characters of string q, which doesn't equal to either string "zyx", "xzy", "yxz". If q doesn't contain any such subsequence, terminate the algorithm, otherwise go to step 2.
2. Rearrange the letters of the found subsequence randomly and go to step 1.

Sereja thinks that the algorithm works correctly on string q if there is a non-zero probability that the algorithm will be terminated. But if the algorithm anyway will work for infinitely long on a string, then we consider the algorithm to work incorrectly on this string.

Sereja wants to test his algorithm. For that, he has string $s = s_1s_2... \ s_n$, consisting of n characters. The boy conducts a series of m tests. As the i-th test, he sends substring $s_{l_i}s_{l_i+1}... \ s_{r_i}$ $(1 \le l_i \le r_i \le n)$ to the algorithm input. Unfortunately, the implementation of his algorithm works too long, so Sereja asked you to help. For each test (l_i, r_i) determine if the algorithm works correctly on this test or not.

Input

The first line contains non-empty string s, its length (n) doesn't exceed 10^5. It is guaranteed that string s only contains characters: 'x', 'y', 'z'.

The second line contains integer m $(1 \le m \le 10^5)$ — the number of tests. Next m lines contain the tests. The i-th line contains a pair of integers l_i, r_i $(1 \le l_i \le r_i \le n)$.

Output

For each test, print "YES" (without the quotes) if the algorithm works correctly on the corresponding test and "NO" (without the quotes) otherwise.

Examples

input

```
zyxxxxxxyyz
5
5 5
1 3
1 11
1 4
3 6
```

output

```
YES
YES
NO
YES
NO
```

Note

In the first example, in test one and two the algorithm will always be terminated in one step. In the fourth test you can get string "xzyx" on which the algorithm will terminate. In all other tests the algorithm doesn't work correctly.

B. Sereja ans Anagrams

time limit per test: 1 second

memory limit per test: 256 megabytes

input: standard input

output: standard output

Sereja has two sequences a and b and number p. Sequence a consists of n integers $a_1, a_2, \ldots, a_n$. Similarly, sequence b consists of m integers $b_1, b_2, \ldots, b_m$. As usual, Sereja studies the sequences he has. Today he wants to find the number of positions q $(q + (m - 1) \cdot p \le n;\ q \ge 1)$, such that sequence b can be obtained from sequence $a_q, a_{q+p}, a_{q+2p}, \ldots, a_{q+(m-1)p}$ by rearranging elements.

Sereja needs to rush to the gym, so he asked to find all the described positions of q.

Input

The first line contains three integers n, m and p $(1 \le n, m \le 2 \cdot 10^5, 1 \le p \le 2 \cdot 10^5)$. The next line contains n integers a_1, a_2, $\ldots$, a_n $(1 \le a_i \le 10^9)$. The next line contains m integers b_1, b_2, $\ldots$, b_m $(1 \le b_i \le 10^9)$.

Output

In the first line print the number of valid qs. In the second line, print the valid values in the increasing order.

Examples

input

```
5 3 1
1 2 3 2 1
1 2 3
```

output

```
2
1 3
```

input

```
6 3 2
1 3 2 2 3 1
1 2 3
```

output

```
2
1 2
```

C. Sereja and the Arrangement of Numbers

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

Let's call an array consisting of n integer numbers a_1, a_2, $\ldots$, a_n, *beautiful* if it has the following property:

- consider all pairs of numbers x, y $(x \neq y)$, such that number x occurs in the array a and number y occurs in the array a;
- for each pair x, y must exist some position j $(1 \leq j < n)$, such that at least one of the two conditions are met, either $a_j = x, a_{j+1} = y$, or $a_j = y, a_{j+1} = x$.

Sereja wants to build a beautiful array a, consisting of n integers. But not everything is so easy, Sereja's friend Dima has m coupons, each contains two integers q_i, w_i. Coupon i costs w_i and allows you to use as many numbers q_i as you want when constructing the array a. Values q_i are distinct. Sereja has no coupons, so Dima and Sereja have made the following deal. Dima builds some beautiful array a of n elements. After that he takes w_i rubles from Sereja for each q_i, which occurs in the array a. Sereja believed his friend and agreed to the contract, and now he is wondering, what is the maximum amount of money he can pay.

Help Sereja, find the maximum amount of money he can pay to Dima.

Input

The first line contains two integers n and m $(1 \leq n \leq 2 \cdot 10^6, 1 \leq m \leq 10^5)$. Next m lines contain pairs of integers. The i-th line contains numbers q_i, w_i $(1 \leq q_i, w_i \leq 10^5)$.

It is guaranteed that all q_i are distinct.

Output

In a single line print maximum amount of money (in rubles) Sereja can pay.

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

Examples

input

```
5 2
1 2
2 3
```

output

```
5
```

input

```
100 3
1 2
2 1
3 1
```

output

```
4
```

input

```
1 2
1 1
2 100
```

output

```
100
```

Note

In the first sample Sereja can pay 5 rubles, for example, if Dima constructs the following array: $[1, 2, 1, 2, 2]$. There are another optimal arrays for this test.

In the third sample Sereja can pay 100 rubles, if Dima constructs the following array: $[2]$.

D. Sereja and Sets

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

Sereja has m non-empty sets of integers $A_1, A_2, ..., A_m$. What a lucky coincidence! The given sets are a partition of the set of all integers from 1 to n. In other words, for any integer v $(1 \le v \le n)$ there is exactly one set A_t such that $v \in A_t$. Also Sereja has integer d.

Sereja decided to choose some sets from the sets he has. Let's suppose that $i_1, i_2, ..., i_k$ $(1 \le i_1 < i_2 < ... < i_k \le m)$ are indexes of the chosen sets. Then let's define an array of integers b, sorted in ascending order, as a union of the chosen sets, that is, $A_{i_1} \cup A_{i_2} \cup \cdots \cup A_{i_k}$. We'll represent the element with number j in this array (in ascending order) as b_j. Sereja considers his choice of sets *correct*, if the following conditions are met:

$$b_1 \le d;\ b_{i+1} - b_i \le d\ (1 \le i < |b|);\ n - d + 1 \le b_{|b|}.$$

Sereja wants to know what is the minimum number of sets (k) that he can choose so that his choice will be correct. Help him with that.

Input

The first line contains integers n, m, d $(1 \le d \le n \le 10^5, 1 \le m \le 20)$. The next m lines contain sets. The first number in the i-th line is s_i $(1 \le s_i \le n)$. This number denotes the size of the i-th set. Then the line contains s_i distinct integers from 1 to n — set A_i.

It is guaranteed that the sets form partition of all integers from 1 to n.

Output

In a single line print the answer to the problem — the minimum value k at the right choice.

Examples

input

```
3 2 2
1 2
2 1 3
```

output

```
1
```

input

```
5 1 1
5 4 5 3 2 1
```

output

```
1
```

input

```
7 3 1
4 1 3 5 7
2 2 6
1 4
```

output

```
3
```

E. Sereja and Intervals

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

Sereja is interested in intervals of numbers, so he has prepared a problem about intervals for you. An interval of numbers is a pair of integers $[l, r]$ $(1 \le l \le r \le m)$. Interval $[l_1, r_1]$ belongs to interval $[l_2, r_2]$ if the following condition is met: $l_2 \le l_1 \le r_1 \le r_2$.

Sereja wants to write out a sequence of n intervals $[l_1, r_1]$, $[l_2, r_2]$, ..., $[l_n, r_n]$ on a piece of paper. At that, no interval in the sequence can belong to some other interval of the sequence. Also, Sereja loves number x very much and he wants some (at least one) interval in the sequence to have $l_i = x$. Sereja wonders, how many distinct ways to write such intervals are there?

Help Sereja and find the required number of ways modulo 1000000007 $(10^9 + 7)$.

Two ways are considered distinct if there is such j $(1 \le j \le n)$, that the j-th intervals in two corresponding sequences are not equal.

Input

The first line contains integers n, m, x $(1 \le n \cdot m \le 100000, 1 \le x \le m)$ — the number of segments in the sequence, the constraints on the numbers in segments and Sereja's favourite number.

Output

In a single line print the answer modulo 1000000007 $(10^9 + 7)$.

Examples

input
```
1 1 1
```
output
```
1
```

input
```
3 5 1
```
output
```
240
```

input
```
2 3 3
```
output
```
6
```

Note

In third example next sequences will be correct: $\{[1, 1], [3, 3]\}$, $\{[1, 2], [3, 3]\}$, $\{[2, 2], [3, 3]\}$, $\{[3, 3], [1, 1]\}$, $\{[3, 3], [2, 2]\}$, $\{[3, 3], [1, 2]\}$.

Codeforces (c) Copyright 2010-2016 Mike Mirzayanov
The only programming contests Web 2.0 platform