A. Levko and Array Recovery

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

Levko loves array $a_1, a_2, \dots, a_n$, consisting of integers, very much. That is why Levko is playing with array a, performing all sorts of operations with it. Each operation Levko performs is of one of two types:

1. Increase all elements from l_i to r_i by d_i. In other words, perform assignments $a_j = a_j + d_i$ for all j that meet the inequation $l_i \le j \le r_i$.
2. Find the maximum of elements from l_i to r_i. That is, calculate the value $m_i = \max_{j=l_i}^{r_i} a_j$.

Sadly, Levko has recently lost his array. Fortunately, Levko has records of all operations he has performed on array a. Help Levko, given the operation records, find at least one suitable array. The results of all operations for the given array must coincide with the record results. Levko clearly remembers that all numbers in his array didn't exceed 10^9 in their absolute value, so he asks you to find such an array.

Input

The first line contains two integers n and m ($1 \le n, m \le 5000$) — the size of the array and the number of operations in Levko's records, correspondingly.

Next m lines describe the operations, the i-th line describes the i-th operation. The first integer in the i-th line is integer t_i ($1 \le t_i \le 2$) that describes the operation type. If $t_i = 1$, then it is followed by three integers l_i, r_i and d_i ($1 \le l_i \le r_i \le n$, $-10^4 \le d_i \le 10^4$) — the description of the operation of the first type. If $t_i = 2$, then it is followed by three integers l_i, r_i and m_i ($1 \le l_i \le r_i \le n$, $-5 \cdot 10^7 \le m_i \le 5 \cdot 10^7$) — the description of the operation of the second type.

The operations are given in the order Levko performed them on his array.

Output

In the first line print "YES" (without the quotes), if the solution exists and "NO" (without the quotes) otherwise.

If the solution exists, then on the second line print n integers $a_1, a_2, \dots, a_n$ ($|a_i| \le 10^9$) — the recovered array.

Examples

input

```
4 5
1 2 3 1
2 1 2 8
2 3 4 7
1 1 3 3
2 3 4 8
```

output

```
YES
4 7 4 7
```

input

```
4 5
1 2 3 1
2 1 2 8
2 3 4 7
1 1 3 3
2 3 4 13
```

output

```
NO
```

B. Levko and Array

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

Levko has an array that consists of integers: $a_1, a_2, \dots, a_n$. But he doesn’t like this array at all.

Levko thinks that the beauty of the array a directly depends on value $c(a)$, which can be calculated by the formula:

$$c(a) = \max_{1 \le i \le n-1} |a_{i+1} - a_i|, n > 1; c(a) = 0, 0 \le n \le 1.$$

The less value $c(a)$ is, the more beautiful the array is.
It’s time to change the world and Levko is going to change his array for the better. To be exact, Levko wants to change the values of at most k array elements (it is allowed to replace the values by any integers). Of course, the changes should make the array as beautiful as possible.

Help Levko and calculate what minimum number $c(a)$ he can reach.

Input

The first line contains two integers n and k ($1 \le k \le n \le 2000$). The second line contains space-separated integers $a_1, a_2, \dots, a_n$ ($-10^9 \le a_i \le 10^9$).

Output

A single number — the minimum value of $c(a)$ Levko can get.

Examples

input

```
5 2
4 7 4 7 4
```

output

```
0
```

input

```
3 1
-100 0 100
```

output

```
100
```

input

```
6 3
1 2 3 7 8 9
```

output

```
1
```

Note

In the first sample Levko can change the second and fourth elements and get array: $4, 4, 4, 4, 4$.

In the third sample he can get array: $1, 2, 3, 4, 5, 6$.

C. Levko and Strings

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

Levko loves strings of length n, consisting of lowercase English letters, very much. He has one such string s. For each string t of length n, Levko defines its beauty relative to s as the number of pairs of indexes i, j $(1 \le i \le j \le n)$, such that substring **$t[i..j]$ is lexicographically larger than substring $s[i..j]$**.

The boy wondered how many strings t are there, such that their beauty relative to s equals exactly k. Help him, find the remainder after division this number by 1000000007 $(10^9 + 7)$.

A substring $s[i..j]$ of string $s = s_1s_2... s_n$ is string $s_is_{i+1}... s_j$.

String $x = x_1x_2... x_p$ is lexicographically larger than string $y = y_1y_2... y_p$, if there is such number r $(r < p)$, that $x_1 = y_1, x_2 = y_2, ... , x_r = y_r$ and $x_{r+1} > y_{r+1}$. The string characters are compared by their ASCII codes.

Input

The first line contains two integers n and k $(1 \le n \le 2000, 0 \le k \le 2000)$.

The second line contains a non-empty string s of length n. String s consists only of lowercase English letters.

Output

Print a single number — the answer to the problem modulo 1000000007 $(10^9 + 7)$.

Examples

input

```
2 2
yz
```

output

```
26
```

input

```
2 3
yx
```

output

```
2
```

input

```
4 7
abcd
```

output

```
21962
```

D. Levko and Sets

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

Levko loves all sorts of sets very much.

Levko has two arrays of integers $a_1, a_2, \ldots, a_n$ and $b_1, b_2, \ldots, b_m$ and a prime number p. Today he generates n sets. Let's describe the generation process for the i-th set:

1. First it has a single number 1.
2. Let's take any element c from this set. For all j ($1 \le j \le m$) if number $(c \cdot a_i^{b_j}) \mod p$ doesn't occur in the set, then add it to the set.
3. Repeat step 2 as long as we can add at least one element to our set.

Levko wonders, how many numbers belong to at least one set. That is, he wants to know what size is the union of n generated sets.

Input

The first line contains three integers n, m and p ($1 \le n \le 10^4$, $1 \le m \le 10^5$, $2 \le p \le 10^9$), p is prime.

The second line contains space-separated integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i < p$). The third line contains space-separated integers $b_1, b_2, \ldots, b_m$ ($1 \le b_i \le 10^9$).

Output

The single number — the size of the union of the sets.

Examples

input

```
1 1 7
2
5
```

output

```
3
```

input

```
1 2 7
2
2 4
```

output

```
3
```

input

```
2 1 7
1 6
2
```

output

```
1
```

input

```
2 1 7
1 6
5
```

output

```
2
```

E. Levko and Game

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

Levko loves sports pathfinding competitions in his city very much. In order to boost his performance, Levko spends his spare time practicing. The practice is a game.

The city consists of n intersections connected by $m+k$ directed roads. Two or more roads can connect the same pair of intersections. Besides, there can be roads leading from an intersection to itself.

Levko and Zenyk are playing a game. First Levko stands on intersection s_1, and Zenyk stands on intersection s_2. They both want to get to intersection f. The person who does it quicker wins. If they get there at the same time, the game ends with a draw. By agreement both players start simultaneously and move with the same speed.

Levko wants to win very much. He knows the lengths of all the roads in the city. Also he knows that he can change the lengths of some roads (there are k such roads at all) if he pays the government. So, the government can change the length of the i-th road to any integer value in the segment $[l_i, r_i]$ (both borders inclusive). Levko wondered if he can reconstruct the roads so as to win the game and whether he can hope for the draw if he cannot win.

You should consider that both players play optimally well. It is guaranteed that we can get from intersections s_1 and s_2 to intersection f.

Input

The first line contains three integers n, m and k ($1 \le n, m \le 10^4$, $1 \le k \le 100$). The second line contains three integers s_1, s_2 and f ($1 \le s_1, s_2, f \le n$).

The next m lines contains the descriptions of the roads that cannot be changed by Levko. Each line contains three integers a_i, b_i and c_i ($1 \le a_i, b_i \le n, 1 \le c_i \le 10^9$), representing a road from intersection a_i to intersection b_i of length c_i.

The next k lines contains the descriptions of the roads that can be changed by Levko. Each line contains four integers a_i, b_i, l_i and r_i ($1 \le a_i, b_i \le n, 1 \le l_i \le r_i \le 10^9$), representing a road from intersection a_i to intersection b_i, Levko can set the road's length within limits $[l_i, r_i]$.

Consider all intersections numbered from 1 to n. It is guaranteed that you can get from intersections s_1 and s_2 to intersection f.

Output

In the first line print string "WIN" (without the quotes) if Levko can win this game, string "DRAW" (without the quotes) if Levko can end the game with a draw and "LOSE" (without the quotes) if he loses for sure.

If the answer is "WIN" or "DRAW", then print on the second line k space-separated integers — the length of the roads Levko sets in the order they occur in the input.

Examples

input

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

output

```
WIN
1 1 3
```

input

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

output

```
DRAW
1 1 2
```

input

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

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

output

```
LOSE
```