C. Functions

Time Limit: 10 seconds
Memory Limit: 2 GB

You are given a sequence of functions. At position i, the function f_i is defined as:

$$f_i(x) = \begin{cases} (x - d_i) \bmod a_i, & \text{if } x \in [l_i, r_i] \\ x, & \text{otherwise} \end{cases}$$

For function i, if the input x satisfies both $x \in [l_i, r_i]$ and $x - d_i \geq a_i$, we say that the function is activated.

Each query gives you three integers L, R, and x. Starting from the initial value x, apply all functions in the range $[L, R]$ sequentially, updating $x \leftarrow f_i(x)$ after each function. For every activation that occurs during this process, compute the bitwise XOR of the corresponding a_i values.

Return the final XOR result for each query.

The queries are given in an **online** manner — you must compute the answer to the previous query before you can determine the parameters of the next one.

Input Format

The first line contains two integers n and m, representing the number of functions and the number of queries.

The next n lines each contain four integers l_i, r_i, d_i, and a_i, describing the function at position i as defined previously.

The following m lines each represent a query, given as three integers:

$$L \oplus (z \bmod n), \quad R \oplus (z \bmod n), \quad x \oplus z$$

Here, z is the result of the previous query (with $z = 0$ for the first query), and $\oplus$ denotes bitwise XOR.

Output Format

For each query, output a single line containing one integer — the XOR of all a_i values corresponding to activations during the function applications.

Sample 1

input

```
10 10
2 2 2 2
1 2 0 1
1 1 0 1
0 0 0 1
2 2 0 1
3 3 0 1
22 22 18 14
5 5 1 4
1 1 0 1
1 1 0 1
2 10 73
3 9 70
1 9 71
1 5 71
1 10 8
1 10 67
1 10 5
5 14 93
3 10 14
3 10 88
```

output

```
0
0
0
0
0
0
4
0
0
0
```

Sample 2

input

```
20 20
1 1 0 1
18 23 8 11
5 9 4 4
1 2 1 1
37 38 18 19
3 3 2 1
14 18 5 9
1 1 0 1
2 3 2 1
2 2 1 1
1 1 0 1
1 1 1 1
3 3 1 1
1 1 0 1
15 15 8 7
1 1 0 1
1 1 0 1
1 2 1 1
1 2 1 1
33 65 3 32
1 15 18
1 20 79
1 19 8
13 16 39
4 24 56
2 20 75
4 10 1
8 13 40
2 20 87
1 20 93
1 2 69
2 19 84
1 20 35
15 30 4
10 16 70
2 20 1
1 16 57
2 20 54
```

```
14 31 50
6 20 53
```

output

```
0
0
4
32
0
0
0
0
0
0
0
0
32
0
0
0
0
32
0
32
```

Samples 3–7

See attached files.

Constraints

- For 5% of the points: $1 \leq n, m \leq 10^3$
- For another 10%: $d_i = 0$ for all functions
- For another 10%: every query satisfies $L = 1$ and $R = n$
- For another 10%: every query satisfies $L = 1$
- For another 10%: every query satisfies $R = n$
- For another 10%: $d_i, l_i, r_i, a_i, x \leq 10^9$
- For another 10%: $1 \leq n \leq 3 \times 10^4$, $1 \leq m \leq 2 \times 10^5$
- For another 10%: $1 \leq n \leq 5 \times 10^4$, $1 \leq m \leq 3 \times 10^5$
- For another 10%: $1 \leq n \leq 7 \times 10^4$, $1 \leq m \leq 4 \times 10^5$

- For all test cases:
  - $1 \leq n \leq 10^5$
  - $1 \leq m \leq 5 \times 10^5$
  - $0 \leq d_i \leq l_i \leq r_i \leq 10^{18}$
  - $1 \leq x, a_i \leq 10^{18}$
  - $1 \leq L \leq R \leq n$

Note

The queries are encoded using XOR:

- $L \oplus (z \bmod n)$
- $R \oplus (z \bmod n)$
- $x \oplus z$

Where z is the answer from the previous query ($z = 0$ for the first one). The XOR result might exceed the original bounds of L, R, or z. The constraint ranges of L and R refer to the values **after** decoding.