Distance and Union

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	512 megabytes

You are given an undirected graph with n vertices. Initially it does not contain any edges.

You should perform two types of queries with this graph:

1. Add an edge between vertices u_i and v_i. It is guaranteed that there was no such edge in a graph before the query and that after performing the query graph is a forest.
2. Print the number of vertices w such that the distance between u_i and w equals to k_i where the distance is the number of edges in a simple path connecting u_i and w.

Input

The first line contains two integers n and q ($1 \le n \le 100\,000$, $1 \le q \le 200\,000$), the number of vertices and the number of queries respectively.

Next q lines contain three integers t_i, a_i, b_i ($1 \le t_i \le 2$, $0 \le a_i, b_i \le n - 1$), the type of query and two numbers describing the query.

Let $last$ be the answer to the last query of second type or zero if there were no queries of second type yet. Then, if $t_i = 1$, you need to add an edge between vertices u_i and v_i where $u_i = ((a_i + last) \mod n) + 1$ and $v_i = ((b_i + last) \mod n) + 1$. If $t_i = 2$, you need to calculate the number of vertices w such that the distance between u_i and w is k_i where $u_i = ((a_i + last) \mod n) + 1$ and $k_i = (b_i + last) \mod n$.

Output

Print all answers to the queries of the second type in separate lines.

Example

standard input	standard output
6 7 1 0 1 1 2 3 1 2 4 2 2 1 1 0 3 1 5 0 2 5 0	2 3

Note

In the sample test the performed queries are:

1. Add an edge between 1 and 2.
2. Add an edge between 3 and 4.
3. Add an edge between 3 and 5.
4. Find the number of vertices w such that the distance between 3 and w equals to 1. There are two such vertices: 4 and 5; $last$ becomes equal to 2.
5. Add an edge between 3 and 6.

6. Add an edge between 2 and 3.

7. Find the number of vertices w such that the distance between 2 and w equals to 2. There are three such vertices: 4, 5 and 6; *last* becomes equal to 3.