Locks

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

There are n elements $a_1, a_2, \ldots, a_n$, each with a lock. Initially, all elements are unlocked.

Given q queries, the i-th query consists of four integers (t_s, t_e, l, r), where $1 \le l \le r \le n$. Each query performs a locking process and an unlocking process on the timeline.

Locking process: Starting from time t_s, in the next $r - l + 1$ time instants, the elements $a_l, a_{l+1}, \ldots, a_r$ are processed in order. Specifically, at time $t_s + k$, element a_{l+k} is processed $(0 \le k \le r - l)$. If this element is currently unlocked, it is locked, and the lock is marked as belonging to the current query; if it is already locked by another query, no operation is performed. That is, if when accessing this lock, it has already been locked by another locking process, this lock will not have any changes.

Unlocking process: Starting from time t_e, in the next $r-l+1$ time instants, the elements $a_l, a_{l+1}, \ldots, a_r$ are processed in order. Specifically, at time $t_e + k$, element a_{l+k} is processed $(0 \le k \le r - l)$. If the current lock on this element was added by this query during the locking process, it is unlocked; otherwise, no operation is performed.

For all operations occurring at the same time on the same element, they are executed in ascending order of query index.

For each query, count the number of elements that were **successfully locked** during its locking process.

Input

The first line contains two integers n, q $(1 \le n, q \le 1000)$.

The next q lines each contain four integers t_s, t_e, l, r $(1 \le l \le r \le n, 1 \le t_s < t_e \le 10^9)$, representing a query.

Output

Output q lines, where the i-th line contains the number of elements successfully locked by the i-th query.

Examples

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