Luggage

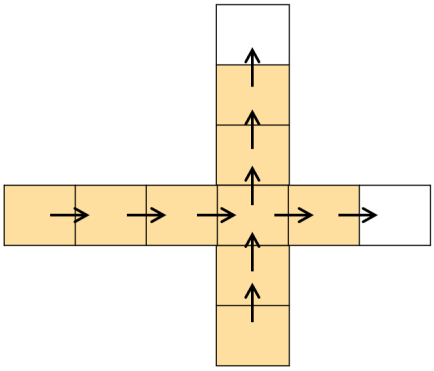
Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	512 megabytes

Please note the non-standard scoring system.

After working on the New Year patterns, Amir and Ilnur noticed that lines intersect not only on a laptop screen. Renata told them about the baggage handling system at an airport: its conveyor belts also meet at a single point. Unlike the intersections in their drawings, however, these intersections can cause bags to collide!

The airport has k baggage conveyor belts. All belts move continuously at the same speed: a bag travels one unit of distance per second.

The length of the i-th belt is d_i. All belts pass through a common sorting junction, located at a distance of d'_i from the beginning of the i-th belt. Thus, the distance from the junction to the end of that belt is $d_i - d'_i$.



An example of two conveyor belts.

The system must deliver n bags. For each bag i, the following parameters are given:

- $time_i$ — the time at which a robot arrives with the bag at the beginning of a conveyor belt;
- $start_i$ — the index of the conveyor belt onto which the robot must place the bag;
- $finish_i$ — the index of the conveyor belt at whose end the bag must be delivered.

The robot may place the bag at the beginning of belt $start_i$ immediately at time $time_i$, or wait for a non-negative integer number of seconds. Denote this waiting time by $idle_time_i$.

Once the robot has placed the bag on a belt, the bag cannot be stopped or delayed. It travels along belt $start_i$ to the sorting junction, where it is automatically redirected towards the end of belt $finish_i$.

Redirection takes no additional time. In particular, if $start_i = finish_i$, the bag simply continues along the same belt.

If a bag reaches the sorting junction at time t, it occupies the junction for exactly one second, that is, during the half-open interval $[t, t + 1)$.

At most one bag may occupy the sorting junction at any given time.

For example, one bag may reach the junction at time 5 and another at time 6, but two bags cannot both reach it at time 5.

Collisions outside the sorting junction are not considered.

For bag i with waiting time $idle_time_i$:

- the time at which it reaches the sorting junction is $t_i = time_i + idle_time_i + d'_{start_i}$;
- the time at which it is delivered to the end of the required belt is $t_i + d_{finish_i} - d'_{finish_i}$.

The one second during which a bag occupies the sorting junction is already included in its travel time from the junction to the end of the belt; it must not be added separately.

Let end_time be the time at which the last bag is delivered, i.e., the maximum delivery time over all bags.

Choose $idle_time_i$ for every bag, satisfying the following objectives in the specified order of priority:

1. Ensure that no two bags collide at the sorting junction.
2. Among all collision-free schedules, minimize end_time.
3. Among all schedules with minimum end_time, minimize the maximum waiting time of any robot: $\max_{1 \le i \le n} idle_time_i$.

You do not have to satisfy all three objectives to receive partial points. The scoring rules are described below.

Input

The first line contains two integers n and k — the number of bags and the number of conveyor belts ($2 \le n, k \le 3 \cdot 10^5$).

Each of the next k lines contains two integers d'_i and d_i — the distance from the beginning of belt i to the sorting junction and the total length of the belt, respectively ($1 \le d'_i < d_i \le 10^{18}$).

Each of the next n lines contains three integers $time_i$, $start_i$, and $finish_i$ — the arrival time of the robot, the index of the starting belt, and the index of the destination belt, respectively ($1 \le time_i \le 10^{18}$; $1 \le start_i, finish_i \le k$).

The bags are listed in non-decreasing order of robot arrival times: $time_i \le time_{i+1}$ for all $1 \le i < n$.

Output

Print n integers $idle_time_1, idle_time_2, \ldots, idle_time_n$ on a single line — the waiting times of the robots in seconds ($0 \le idle_time_i \le 2 \cdot 10^{18}$).

Scoring

The tests are divided into eight scored subtasks and subtask 0, which contains the examples from the statement.

Each test is scored as follows:

- 0% — if the answer is incorrect or at least two bags collide at the sorting node;
- 40% — if there are no collisions, but end_time is not minimal;
- 70% — if there are no collisions and end_time is minimal, but the value $\max_i idle_time_i$ is not minimal among such schedules;
- 100% — if there are no collisions and both optimality criteria are satisfied.

The percentage of points for a subtask is equal to the **minimum percentage** obtained on its tests and on the tests of all subtasks listed in the "Required subtasks"column.

A subtask receives the corresponding percentage of the number of points specified in the table. Subtask 0 brings no points.

Subtask	**Points**	**Additional constraints**	**Required subtasks**
0	0	Examples from the statement	—
1	10	$k = 2$; $2d'_1 = d_1 = 2d'_2 = d_2$	—
2	10	$k = 2$; $d'_1 = d'_2$	1
3	10	$k = 2$; $time_1 = time_2$	—
4	10	$k = 2$	0–3
5	20	$time_i = time_j$ for all $i \neq j$	3
6	10	$start_i = finish_i$ for all i	—
7	20	$k \leq 20$	0–4
8	10	No additional constraints.	0–7

Examples

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

Note

Note that the output in the first sample does not receive full points; it receives 70%.

Details for the first output:

Bag number	Arrival time	Waiting time	Junction arrival time	Delivery time
1	1	0	$1+0+2=3$	$3+5-3=5$
2	1	1	$1+1+3=5$	$5+5-2=8$
3	2	0	$2+0+2=4$	$4+5-2=7$
4	3	2	$3+2+2=7$	$7+5-3=9$
5	3	0	$3+0+3=6$	$6+5-3=8$

Details for the second output:

Bag number	Arrival time	Waiting time	Junction arrival time	Delivery time
1	1	0	$1+0+2=3$	$3+5-3=5$
2	1	0	$1+0+3=4$	$4+5-2=7$
3	2	1	$2+1+2=5$	$5+5-2=8$
4	3	1	$3+1+2=6$	$6+5-3=8$
5	3	1	$3+1+3=7$	$7+5-3=9$