E. Buses and People

The main Bertown street is represented by a straight line. There are 10^9 bus stops located on the line. The stops are numbered with integers from 1 to 10^9 in the order in which they follow on the road. The city has n buses. Every day the i-th bus drives from stop number s_i to stop number f_i ($s_i < f_i$), it stops on all intermediate stops and returns only at night. The bus starts driving at time t_i and drives so fast that it finishes driving also at time t_i. The time t_i is different for all buses. The buses have infinite capacity.

Bertown has m citizens. Today the i-th person should get from stop number l_i to stop number r_i ($l_i < r_i$); the i-th citizen comes to his initial stop (l_i) at time b_i. Each person, on the one hand, wants to get to the destination point as quickly as possible, and on the other hand, definitely does not want to change the buses as he rides. More formally: the i-th person chooses bus j, with minimum time t_j, such that $s_j \le l_i$, $r_i \le f_j$ and $b_i \le t_j$.

Your task is to determine for each citizen whether he can ride to the destination point today and if he can, find the number of the bus on which the citizen will ride.

Input

The first line contains two integers n and m ($1 \le n, m \le 10^5$) — the number of buses and the number of people.

Then n lines follow, each of them contains three integers: s_i, f_i, t_i ($1 \le s_i, f_i, t_i \le 10^9$, $s_i < f_i$) — the description of the buses. It is guaranteed that all t_i-s are different.

Then m lines follow, each of them contains three integers: l_i, r_i, b_i ($1 \le l_i, r_i, b_i \le 10^9$, $l_i < r_i$) — the Bertown citizens' description. Some b_i-s could coincide.

Output

In the first line print m space-separated integers: the i-th number should be equal either to -1, if the person number i can't get to the destination point, or to the number of the bus that will ride the person number i. The buses are numbered with integers from 1 to n in the input order.

Examples

input
```
4 3
1 10 10
5 6 2
6 7 3
5 7 4
5 7 1
1 2 1
1 10 11
```

output
```
4 1 -1
```

input
```
1 1
1 1000000000 1000000000
1 1000000000 1000000000
```

output
```
1
```