COCI '11 Contest 2 #6 Raspored

Time limit: 2.0s **Memory limit:** 128M

Mirko's pizza place is the best one in town. It is so good that all town residents eat pizza for lunch every day. Mirko's delivery service is so fast that the delivery time is negligible. The problem is baking the pizzas, since all residents have their own favourite topping combination, so baking pizzas for two different residents doesn't always take the same amount of time. Mirko only has one small baking oven with a capacity of a single pizza at a time, so good scheduling is extremely important and must be determined **before the day starts**.

For each of the N town residents (denoted by numbers from 1 to N), we know the baking duration for their favourite pizza (T_i), as well as the moment in the day when they plan on having lunch (L_i). If a resident receives their pizza K moments before the planned lunch time, Mirko is rewarded with a tip of K kunas[1]. On the other hand, if the pizza is delivered K moments late (after the planned lunch time), Mirko must pay the resident K kunas (because of his timely delivery insurance policy). If the pizza is delivered precisely on time, Mirko won't get a tip, but he doesn't have to pay anything either.

Mirko would like to know the **maximum total tip** (including all insurance payouts as negative tips) that can be earned in a day if pizzas are baked in an optimal order. Notice that Mirko can earn a negative total tip (if he has to pay out more insurance than the amount of tips he receives).

Since residents sometimes change their favourite pizza toppings, as well as their preferred lunch time, Mirko's schedule must be adapted in order to keep earning optimal tip amounts. Write a program to compute the maximum total tip for the beginning requirements, as well as after each change.

Note: In this town, the day starts at the moment $t = 0$ and lasts much longer than the time needed to bake pizzas for all residents. The schedule, including the adaptations, must be determined before the day starts.

[1] Kuna - Croatian currency. 1 kuna = 100 lipa(s); 1 € = about 7.5 kuna(s). Come to Croatia! :)

Input Specification

The first line of input contains two positive integers N and C, the number of residents and the number of pizza requirement changes, respectively.

Each of the next N lines contains two positive integers: L_i, the moment when resident i plans on having lunch, and T_i, the time needed to bake the pizza for resident i.

Each of the next C lines contains three positive integers: R (the index of a resident), L (the new moment when resident R plans on having lunch), and T (the time needed to bake resident R's new favourite pizza).

Constraints

$1 \leq N, C \leq 200\,000$

$0 \leq L_i, L \leq 100\,000$

$1 \leq T_i, T \leq 100\,000$

$1 \leq R \leq N$

Output Specification

The first line of output must contain the maximum total tip for the beginning requirements of the residents.

For each of the C changes, the output must contain an additional line of output containing the new maximum total tip value after the change.

Scoring

In test cases worth 50% of points, the following constraint holds: $1 \leq T_i, T \leq 1000$.

Sample Input 1

```
3 2
10 2
6 5
4 3
1 6 1
3 0 10
```

Sample Output 1

```
3
2
-11
```

Explanation for Sample Output 1

The optimal pizza baking schedule is $(1, 3, 2)$. That way, the first pizza will be finished at the moment $t = 2$, the third one at $t = 5$, and the second one at $t = 10$. The first pizza will be delivered 8 moments early (8 kunas tip), the second one will be 1 moment late (-1 kuna), and the third one will be 4 moments late (-4 kunas), so the total tip is 3 kunas. When the first resident changes requirements, the optimal schedule remains unchanged, while the tips change to 5, 0, and -3, respectively. After the second requirement change, the optimal schedule is $(1, 2, 3)$, while the tips are 5, 0, and -16, respectively.

Sample Input 2

```
4 2
3 2
0 3
4 3
4 1
3 0 4
1 4 5
```

Sample Output 2

```
-8
-13
-18
```

Sample Input 3

```
6 7
17 5
26 4
5 5
12 4
8 1
18 2
3 31 3
4 11 5
4 19 3
5 23 2
6 15 1
5 19 1
3 10 4
```

Sample Output 3

```
27
59
56
69
78
81
82
58
```