F. Frogs and mosquitoes

There are n frogs sitting on the coordinate axis Ox. For each frog two values x_i, t_i are known — the position and the initial length of the tongue of the i-th frog (it is guaranteed that all positions x_i are different). m mosquitoes one by one are landing to the coordinate axis. For each mosquito two values are known p_j — the coordinate of the position where the j-th mosquito lands and b_j — the size of the j-th mosquito. Frogs and mosquitoes are represented as points on the coordinate axis.

The frog can eat mosquito if mosquito is in the same position with the frog or to the right, and the distance between them is not greater than the length of the tongue of the frog.

If at some moment several frogs can eat a mosquito the leftmost frog will eat it (with minimal x_i). After eating a mosquito the length of the tongue of a frog increases with the value of the size of eaten mosquito. It's possible that after it the frog will be able to eat some other mosquitoes (the frog should eat them in this case).

For each frog print two values — the number of eaten mosquitoes and the length of the tongue after landing all mosquitoes and after eating all possible mosquitoes by frogs.

Each mosquito is landing to the coordinate axis only after frogs eat all possible mosquitoes landed before. Mosquitoes are given in order of their landing to the coordinate axis.

Input

First line contains two integers n, m ($1 \le n, m \le 2 \cdot 10^5$) — the number of frogs and mosquitoes.

Each of the next n lines contains two integers x_i, t_i ($0 \le x_i, t_i \le 10^9$) — the position and the initial length of the tongue of the i-th frog. It is guaranteed that all x_i are different.

Next m lines contain two integers each p_j, b_j ($0 \le p_j, b_j \le 10^9$) — the position and the size of the j-th mosquito.

Output

Print n lines. The i-th line should contain two integer values c_i, l_i — the number of mosquitoes eaten by the i-th frog and the length of the tongue of the i-th frog.

Examples

input
4 6
10 2
15 0
6 1
0 1
110 10
1 1
6 0
15 10
14 100
12 2

output
3 114
1 10
1 1
1 2

input

```
1 2
10 2
20 2
12 1
```

output

```
1 3
```