E. Polycarpus and Tasks

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Polycarpus has many tasks. Each task is characterized by three integers l_i, r_i and t_i. Three integers (l_i, r_i, t_i) mean that to perform task i, one needs to choose an integer s_i $(l_i \le s_i; s_i + t_i - 1 \le r_i)$, then the task will be carried out continuously for t_i units of time, starting at time s_i and up to time $s_i + t_i - 1$, inclusive. In other words, a task is performed for a continuous period of time lasting t_i, should be started no earlier than l_i, and completed no later than r_i.

Polycarpus's tasks have a surprising property: for any task j, k (with $j < k$) $l_j < l_k$ and $r_j < r_k$.

Let's suppose there is an ordered set of tasks A, containing $|A|$ tasks. We'll assume that $a_j = (l_j, r_j, t_j)$ $(1 \le j \le |A|)$. Also, we'll assume that the tasks are ordered by increasing l_j with the increase in number.

Let's consider the following recursive function f, whose argument is an ordered set of tasks A, and the result is an integer. The function $f(A)$ is defined by the greedy algorithm, which is described below in a pseudo-language of programming.

- Step 1. , $ans = 0$.
- Step 2. We consider all tasks in the order of increasing of their numbers in the set A. Lets define the current task counter $i = 0$.
- Step 3. Consider the next task: $i = i + 1$. If $i > |A|$ fulfilled, then go to the 8 step.
- Step 4. If you can get the task done starting at time s_i = max($ans + 1, l_i$), then do the task i: s_i = max($ans + 1, l_i$), $ans = s_i + t_i - 1$, . Go to the next task (step 3).
- Step 5. Otherwise, find such task , that first, task a_i can be done at time s_i = max, and secondly, the value of is positive and takes the maximum value among all b_k that satisfy the first condition. If you can choose multiple tasks as b_k, choose the one with the maximum number in set A.
- Step 6. If you managed to choose task b_k, then , . Go to the next task (step 3).
- Step 7. If you didn't manage to choose task b_k, then skip task i. Go to the next task (step 3).
- Step 8. Return ans as a result of executing $f(A)$.

Polycarpus got entangled in all these formulas and definitions, so he asked you to simulate the execution of the function f, calculate the value of $f(A)$.

Input

The first line of the input contains a single integer n $(1 \le n \le 10^5)$ — the number of tasks in set A.

Then n lines describe the tasks. The i-th line contains three space-separated integers l_i, r_i, t_i $(1 \le l_i \le r_i \le 10^9$, $1 \le t_i \le r_i - l_i + 1)$ — the description of the i-th task.

It is guaranteed that for any tasks j, k (considering that $j < k$) the following is true: $l_j < l_k$ and $r_j < r_k$.

Output

For each task i print a single integer — the result of processing task i on the i-th iteration of the cycle (step 3) in function $f(A)$. In the i-th line print:

- 0 — if you managed to add task i on step 4.
- -1 — if you didn't manage to add or replace task i (step 7).
- res_i $(1 \le res_i \le n)$ — if you managed to replace the task (step 6): res_i equals the task number (in set A), that should be chosen as b_k and replaced by task a_i.

Examples

input

```
5
1 8 5
2 9 3
3 10 3
8 11 4
11 12 2
```

output

```
0 0 1 0 -1
```

input

```
13
1 8 5
2 9 4
3 10 1
4 11 3
8 12 5
9 13 5
10 14 5
11 15 1
12 16 1
13 17 1
14 18 3
15 19 3
16 20 2
```

output

```
0 0 0 2 -1 -1 0 0 0 0 7 0 12
```