

You have a long list of tasks that you need to do today. To accomplish task i you need M_i minutes, and the deadline for this task is D_i . You need not complete a task at a stretch. You can complete a part of it, switch to another task, and then switch back.

You've realized that it might not be possible to complete all the tasks by their deadline. So you decide to do them in such a manner that the maximum amount by which a task's completion time overshoots its deadline is minimized.

Input Format

The first line contains the number of tasks, T . Each of the next T lines contains two integers, D_i and M_i .

Constraints

$$1 \leq T \leq 10^5$$
$$1 \leq D_i \leq 10^5$$
$$1 \leq M_i \leq 10^3$$

Output Format

Output T lines. The i^{th} line contains the value of the maximum amount by which a task's completion time overshoots its deadline, when the first i tasks on your list are scheduled optimally. See the sample input for clarification.

Sample Input

```
5
2 2
1 1
4 3
10 1
2 1
```

Sample Output

```
0
1
2
2
3
```

Explanation

The first task alone can be completed in 2 minutes, and so you won't overshoot the deadline.

With the first two tasks, the optimal schedule can be:
time 1: task 2

time 2: task 1
time 3: task 1

We've overshot task 1 by 1 minute, hence returning 1.

With the first three tasks, the optimal schedule can be:

time 1 : task 2
time 2 : task 1
time 3 : task 3
time 4 : task 1
time 5 : task 3
time 6 : task 3

Task 1 has a deadline 2, and it finishes at time 4. So it exceeds its deadline by 2.

Task 2 has a deadline 1, and it finishes at time 1. So it exceeds its deadline by 0.

Task 3 has a deadline 4, and it finishes at time 6. So it exceeds its deadline by 2.

Thus, the maximum time by which you overshoot a deadline is 2. No schedule can do better than this.

Similar calculation can be done for the case containing 5 tasks.