2823 Harmonic Periods

In real-time scheduling, predictability is very important, i.e., we would like to know the whole schedule before we really run the tasks. Rate-monotonic scheduling is very popular in real-time scheduling for periodic tasks, where tasks with shorter periods have higher priority. However, it is still difficult to know the start time and finish time of each task and they might be different in each period, especially for tasks with low priority, i.e. long period. The hyperperiod, the least common multiple of all periods, is usually too big to be practical to describe the whole schedule. However, if the task periods are harmonic, i.e. are multiples, it is possible to find the start time and finish time of each task quickly because the schedule becomes more regular.

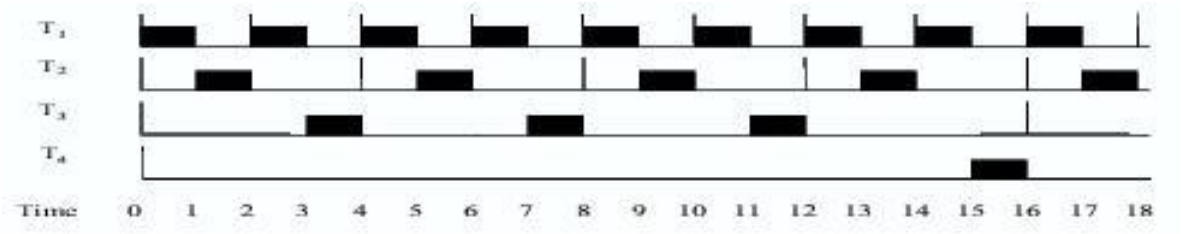

Figure 5: task schedule example

Figure 5 shows that periodic tasks T_1, T_2, T_3, T_4 with execution times 1, 1, 3, 1 and periods 2, 4, 16, 32 respectively are schedulable, each task finishes execution in its period, using the Rate-Monotonic scheduling algorithm since T_1, T_2, T_3, T_4 finish execution at time 1, 2, 12, 16 respectively. T_3 is preempted by T_1 and T_2 at time 4 and time 8 and resume at time 7 and time 11.

Input

All the input numbers are positive integers, < 500000, separated by a space or new line. The first line is the number of task sets. Then, the task sets are listed set by set. Each task set is listed by a line of the number of tasks, ≤ 100, and lines of task *execution time* and *period* pairs (*execution time* $<$ *period*). The periods are harmonic, not sorted, and are different in a task set.

Output

For each task set, find and print out the finish time of the task with the largest period using rate-monotonic scheduling, if schedulable; otherwise print out '`-1`'.

Sample Input

```
3
4
1 2
1 4
3 16
1 32
3
1 4
4 8
```

```
256 1024
3
1 2
3 8
1 4
```

Sample Output

```
16
1024
-1
```