Computer Science Assignment: Conference Scheduling

You are planning a large conference with many lectures. Lecturers submit their topic for the talk and in their submission they must specify the length of their talk. To be fair, as the conference organizer, you make sure that you schedule the lectures in a first come first served basis - you place the lectures in the order that the requests arrived, in the first room that is free.

You have an unlimited number of rooms you could use (so in theory, you could simultaneously place all lectures in different rooms), but each room costs a fixed amount of money so it's in your best interest to minimize the number of rooms you use for your schedule. Ideally, you could just have each of the lectures occur in one room, sequentially. But, the conference attendees are busy people and they can not necessarily afford to take weeks and weeks of time off. Instead, it is required that the conference finish in a fixed amount of time.

Thus, your goal is to schedule the conference so all lectures complete within a given fixed amount of time, minimizing the number of rooms used in the schedule.

Step One: Determine the Minimum Number of Rooms Necessary

The first portion of the assignment is to determine the minimum number of rooms necessary to schedule the conference within the time allotted. If you just do this part correctly, you will earn a 70% on the assignment. To earn up to 90% on the assignment, after you find the correct minimum number of rooms, you must produce a valid schedule for each lecture according to the scheduling details below, which will produce a deterministic schedule, and then output that schedule. To earn up to 100% on the assignment, you must output the schedule sorted in alphabetical order by name of the lecture.

Scheduling Details

Given that you are using k rooms for scheduling, number the rooms from 0 to k-1, inclusive. Here is how you should schedule each of the lectures:

Consider each lecture in the order the requests were received. Initially, all k rooms are free. Place the next lecture in the queue into the very first room that becomes free. If multiple rooms get free at precisely the same time, place the next lecture to be scheduled in the lowest numbered room of all of those available. Naturally, as a consequence of this algorithm, the first k lectures will be placed in rooms 0 through k-1, inclusive, each starting at time t = 0. This system will produce a single deterministic schedule, meaning that for each input case, once the correct value of k is determined, this algorithm will produce one correct answer.

Once you complete the scheduling, for each lecture you will have a room in which the lecture will be given and a starting time. This is the information you must output for each lecture. To earn full credit, you'll have to output each lecture in alphabetical order of lecture name.

Example Schedule

Consider the following lecture requests, received in this order (lecture followed by time):

```
MATH 1000
BIOLOGY 2000
CHEMISTRY 1500
ENGLISH 1500
HISTORY 1300
PHYSICS 2200
SPANISH 1800
CIVICS 800
```

Let's say that we must complete the conference in 5000 seconds.

If we use three rooms, we get the following schedule:

Room 0 schedules MATH from time 0 to 1000.
Room 1 schedules BIOLOGY from time 0 to 2000.
Room 2 schedules CHEMISTRY from time 0 to 1500.
Room 0 schedules ENGLISH from time 1000 to 2500, since Room 0 frees up first.
Room 2 schedules HISTORY from time 1500 to 2800.
Room 1 schedules PHYSICS from time 2000 to time 4200.
Room 0 schedules SPANISH from time 2500 to 4300.
Room 2 schedules CIVICS from time 2800 to 3600.

To show that 3 is the correct minimum number of rooms, consider attempting to schedule the conference using 2 rooms:

Room 0 schedules MATH from time 0 to 1000.
Room 1 schedules BIOLOGY from time 0 to 2000.
Room 0 schedules CHEMISTRY from time 1000 to 2500.
Room 1 schedules ENGLISH from time 2000 to 3500.
Room 0 schedules HISTORY from time 2500 to 3800.
Room 1 schedules PHYSICS from time 3500 to 5700.
Room 0 schedules SPANISH from time 3800 to 5600.
Room 0 schedules CIVICS from time 5600 to 6400.

Difference in Testing

Unlike the previous assignments, your program will be tested on a single input case, but it will be run multiple times so that we can test different input cases. Thus, in the input format, there will NOT be an integer representing the number of test cases. Your program, when run, should simply process a single input case. When we test your program, we will test it on several different input files (piped into standard input as usual).

Input Format

The first line of input will contain two space separated positive integers: n ($n \leq 10^5$), the number of lectures that must be scheduled for the conference, and t ($t \leq 10^9$) the maximum time allotted for the conference, in seconds. The information for each of the lectures follows, in the order that the lecture submissions were made. The next n lines contain the information about the lectures. Each of these lines contains two space separate pieces of information: the name of the lecture (a string of in between 1 and 19 uppercase letters) and the duration of that lecture (a positive integer less than or equal to 10^4.) in seconds. Each of the names of the events will be distinct so that the output order of the lectures is clearly specified.

Output Format

The first line of output will be a single integer, k, the minimum number of rooms necessary to schedule all of the lectures so that the conference completes within the required time limit. This will be followed by n lines of output, one per lecture, in alphabetical order by lecture name. For each lecture, output the following pieces of information separated by spaces: the name of the lecture, the room that lecture is scheduled for, and the time (in seconds) after the beginning of the conference that the lecture will begin.

Sample Input

```
8 5000
MATH 1000
BIOLOGY 2000
CHEMISTRY 1500
ENGLISH 1500
HISTORY 1300
PHYSICS 2200
SPANISH 1800
CIVICS 800
```

Sample Output

```
3
BIOLOGY 1 0
CHEMISTRY 2 0
CIVICS 2 2800
ENGLISH 0 1000
HISTORY 2 1500
MATH 0 0
PHYSICS 1 2000
SPANISH 0 2500
```

Deliverables

Turn in a single file, *conference.c*, over WebCourses that solves the specified problem. Please do not make any enhancements to your program. Make sure it solves this specified problem exactly.