Computer Science 1 – Program 6
Getting into a Club (Heaps)
Assigned: 11/19/10
Due: 12/1/10 (Wednesday) at 11:55pm (WebCourses time)

The Problem

When you have gone downtown on Friday nights, you've noticed a peculiar behavior: after a club gets to capacity, a line of people forms outside, waiting to get in. Whenever a few people leave the club, the people who are let in aren't necessarily the ones who have been waiting the longest. Instead, it appears as if how important one is, matters. For example, even if an A-list celebrity arrives last, she is the very next person let into the club. In essence, who gets chosen to enter the club next has nothing to do with when one got there, but is only dependent upon how important (priority) that person is.

You are frustrated that you never know how long you'll have to wait in line. You've decided that utilizing your newly gained knowledge of heaps from Computer Science I, you'll write a little computer simulation that will predict just how long you'll have to wait to get in the club.

In order to solve this problem, you'll get several "club" situations. Each of these will have information about what time each individual wants to enter the club, his/her priority, and when he/she end up leaving the club. You will have to determine how long each person in line had to wait to get in and how long they spent in the club.

More than one event may occur at exactly the same time. To avoid ambiguity, please process all people LEAVING the club at the same time first before processing people entering the club at that same time. This means that if someone arrives at a particular minute and there is open capacity at the club at that time, that person immediately goes in and doesn't have to wait. Also, the club will never stay open more than 20 hours in a row.

Input File Specification (input.txt)

The input file has a single positive integer, *n*, on its first line, specifying the number of clubs to which you want access in the input file.

The first line of each club case will have a two positive integers *k (k < 1001)* and *c (c < 201)*, where *k* represents the number of events (people entering or leaving the club) and *c* represents the capacity of the club. The next k lines will have information about each person either entering or leaving the club. The ordering of these lines will be chronological. If more than one event occur at the same time, then all the leaving events will be listed before all of the entering events. When processing this information, first process all of the events in the file in the order given, meaning that you remove the people from the club that need to leave, THEN you add the new people into the priority queue that want to enter. FINALLY, if there is room in the club, remove items from the priority queue one by one until the club fills up or the line outside is empty.

Each of the input lines will have the following format:

```
XX:XX [AM/PM] [E/L] NAME PRIORITY
```

`XX:XX` represents the time (hours, minutes, seconds). Hours will be represented with one digit, if possible. Minutes will always be represented with two digits.

`[AM/PM]` – the appropriate two uppercase letters will be provided to specify the time.

`[E/L]` – E represents wanting to enter the club while L represents the time at which one leaves the club.

`NAME` is the name of the person entering or leaving. It is guaranteed to be unique, all uppercase letters and no longer than 29 letters long. Since the club is empty at the beginning and end of each simulation, you will see each name in a case twice: once for entering and once for leaving.

`PRIORITY` is a positive integer from 1 to 100 that represents how important that person is. The lower the number, the more important that person is. If two people have the same priority to get in, then the person who came into line earlier gets to go. This number will NOT be included if the line in the file is specifying the person leaving. (Since you only need to know about the person's priority once.)

The time associated with E isn't necessarily the time that person gets INTO the club, it's when they start waiting in line to get in. If someone leaves before they get admittance into the club, this means that they waited in line and left from the line because they no longer wanted to go in.

Output Specification

*****NOTE***: You should generate your output to a FILE that you will call "out.txt".**

For each input club, print out a header with the following format:

```
Club #c:
```

where c represents the day of the simulation ($1 \leq c \leq n$).

Follow this with a blank line.

For each person who tried to get in the club, write a summary of their experience. Each summary should be on a single line about a single individual. The order of these summaries should be alphabetical by name. The summaries should be in one of the two following formats:

```
NAME waited X minutes and never got into the club.

NAME waited X minutes and stayed in the club for Y minutes.
```

Each input case should be separated with a blank line as well.

*****WARNING*****

Your program MUST adhere to this EXACT format (spacing capitalization, use of dollar signs, periods, punctuation, etc). The graders will use very large input files, resulting in very large output files. As such, the graders will use text comparison programs to compare your output to the correct output. If, for example, you have two spaces between in the output when there should be only one space, this will show up as an error even through you may have the program correct. You WILL get points off if this is the case, which is why this is being explained in detail. Minimum deduction will be 10% of the grade, as the graders will be forced to go to text editing of your program in order to give you an accurate grade.

Note Even if the wait time is 1 minute or less, we still use the plural "minutes". Make sure you follow the exact output format.

Implementation Restrictions

Information about the people waiting to get into a full club must be stored in a heap. Information about people currently in the club can be stored however you like. Since someone may be waiting in line when they choose to leave and a heap only accommodates deleting the item with the highest priority, you must deal with someone leaving the line without deleting them from the heap. What should happen is that they stay in the heap until they WOULD HAVE entered the club. At this time your program should recognize that this person was supposed to have left already and not actually put them in the club.

Sample Input File

```
2
10 3
10:35 PM E SARAH 5
10:40 PM E DAVE 4
10:45 PM E LESLIE 6
10:50 PM E SEAN 4
11:00 PM E JENNIFER 3
12:45 AM L LESLIE
12:50 AM L JENNIFER
1:43 AM L SARAH
1:45 AM L DAVE
1:45 AM L SEAN
12 2
10:15 PM E ANNA 6
10:20 PM E JOHN 4
10:30 PM L ANNA
10:30 PM E STEVE 7
10:30 PM E ZACK 6
10:31 PM E JACK 5
10:50 PM L STEVE
11:20 PM L JOHN
11:25 PM E BOB 3
12:00 AM L BOB
```

```
12:10 AM L JACK
12:20 AM L ZACK
```

Sample Output

```
Club #1:

DAVE waited 0 minutes and stayed in the club for 185 minutes.
JENNIFER waited 105 minutes and stayed in the club for 5 minutes.
LESLIE waited 0 minutes and stayed in the club for 120 minutes.
SARAH waited 0 minutes and stayed in the club for 188 minutes.
SEAN waited 120 minutes and stayed in the club for 55 minutes.

Club #2:

ANNA waited 0 minutes and stayed in the club for 15 minutes.
BOB waited 35 minutes and never got into the club.
JACK waited 49 minutes and stayed in the club for 50 minutes.
JOHN waited 0 minutes and stayed in the club for 60 minutes.
STEVE waited 20 minutes and never got into the club.
ZACK waited 0 minutes and stayed in the club for 110 minutes.
```

Deliverables

Turn in a single file, *club.c*, over WebCourses that solves the specified problem. Make sure to include ample comments and use good programming style, on top of the requirements that are given in the program description above.