Computer Science I Program #6: Priority Hair Salon (Binary Heap)

Please check Webcourses for the Due Date

Read all the pages before starting to write your code

A new hair salon has opened up near your house and they need a computer science student to help with their scheduling. You figure a couple extra bucks can't hurt and take the job!

The hair salon has several stylists (no more than 10 working at a time). Each customer arrives at a particular time and then gets placed in a line to wait for a stylist. (Each stylist has their own waiting line.) The line in which a customer is placed is based on the following:

1. If the customer has no preference, she gets placed in the line for the stylist with the fewest waiting customers. If there's still a tie (multiple stylists have the same number of customers waiting), then the stylist at the lowest numbered hair-cutting station cuts the customer's hair.

2. If the customer has a preferred stylist AND that stylist is working, she gets placed in her preferred stylist's line, regardless of how big it is. If the customer has a preferred stylist and that stylist is not working, then the customer gets placed according to rule #1.

But as the name suggests, the hair salon for which you work gives priority to certain individuals and doesn't work strictly on a first come first served basis. In particular, each line for each stylist is actually a priority queue instead of a standard queue. Whenever a stylist is free to cut hair, he'll choose the customer in his line with the highest priority. Here is how priority is determined:

1. Each customer has a number of loyalty points. The higher the number of loyalty points, the higher the priority for the customer.

2. If two customers have the same number of loyalty points, and only one of those customers has labeled this stylist (the one whose line she is in) as their preferred stylist, then that customer gets priority.

3. If two customers have the same number of loyal points, and either both or neither prefer the stylist whose line they are in, then the tie is broken by name, in alphabetical order. Thus, "SHELLY" would get preference over "YUAN", if the tie isn't broken by the previous items. All customers will have unique names, so this third rule is guaranteed to completely determine priority between a set of waiting customers.

When a customer arrives at the salon, they provide the following information:

1. The time in minutes, after the salon opens that they arrive (non-negative integer)
2. Their name (uppercase string of letters, length in between 1 and 20, inclusive)
3. The name of their preferred stylist (or "NONE" if they don't have a preferred stylist)
4. Number of loyalty points (non-negative integer)
5. Time of their hair cut, in minutes (positive integer)

The last item is important because it also determines the number of loyalty points earned after the cut. In particular, for a ***m*** minute haircut, a customer earns ***m*/10** loyalty points, using integer division.

Finally, in the case that one person completes a haircut at time T for one stylist, and at that exact time (time T) a person arrives for a haircut in the line for the same stylist, FIRST place this person in the priority queue for the stylist before determining who will fill the next spot for the stylist. (So for example, if Sally finishes her haircut with stylist Jerome at time t = 1000, and at that time Becky and Michael are waiting for a haircut from Jerome but at time t = 1000, Nellie arrives for a haircut, ADD Nellie to the priority queue with Becky and Michael and THEN choose which person's hair Jerome will cut next.

The Problem

Your task will be to write a program that takes in information about arriving customers in the order in which they come (each one will come at a distinct minute after the salon opens to avoid concurrency issues), and prints out a log in the order in which the haircuts finish, summarizing each cut. If two cuts end at the same exact time, then the log for the cut made by the stylist at the lower numbered hair cut station should be printed first.

The Input (to be read from standard input)

Note: Your program will be run multiple times, using different input cases. Your program should just process one input case. This is different than most of the previous programs.

The first line of input will contain two space-separated integers, n ($n \leq 100000$), representing the number of customers, and m ($m \leq 10$), representing the number of stylists working at the salon for the input case. (No stylist will have the name "NONE".)

The following m lines will each contain a single string. The i^{th} ($1 \leq i < m$) of these lines will store the name of the hair stylist at the i^{th} styling station. Each of these strings will consist of in between 1 and 20 upper case letters. Each stylist is ready to cut hair at time t = 0 minutes after the salon opens.

The last n lines of the input case will each contain information about a single customer. The i^{th} ($1 \leq i \leq n$) of these lines will store the following space-separated information: t_i ($0 \leq t_i \leq 10^9$), s_i, p_i, l_i ($0 \leq l_i \leq 10^5$) and m_i ($0 \leq m_i \leq 10^4$), representing the time the customer arrives (in minutes) after the salon opens, the customer's name (a string of in between 1 and 20 uppercase letters that is NOT "NONE", the preferred stylist for this customer (if this string is "NONE", that means there is no preferred stylist, otherwise the string represents the name of the preferred stylist, who may or may not be working), the number of loyalty points the customer currently has, and the amount of time in minutes the haircut the customer is requesting will take.

These lines of input will in strictly chronologically increasing order. Thus, for all distinct pairs of integers i and j with $1 \leq i < j \leq n$, $t_i < t_j$. Each of the customer names will be **unique.**

The Output (to be printed to standard out)
For each customer, print out a single line in the following format:

```
CUSTOMER M L STYLIST
```

Where CUSTOMER is the name of the customer who got their hair cut, M is the number of minutes after the salon opened that the haircut was completed, L is the number of loyalty points the customer has AFTER the haircut and STYLIST is the name of the stylist who cut the customer's hair.
These lines should be ordered by the time after the salon opens that the haircuts complete. If there are two or more haircuts that complete at the same time, print these lines out in order of the haircutting station of the stylists, from smallest to largest.

Sample Input

```
9 3
DAVE
SANDY
BELLA
10 JASON SANDY 50 77
11 SARAH NONE 0 30
15 CYNTHIA MAX 44 81
22 RAVI SANDY 30 42
27 CHEN SANDY 30 99
30 AMY DAVE 5 25
31 WENDY SANDY 31 15
33 BRIAN NONE 100 10
37 MAC SANDY 29 45
```

Sample Output

```
SARAH 41 3 DAVE
AMY 66 7 DAVE
JASON 87 57 SANDY
CYNTHIA 96 52 BELLA
WENDY 102 32 SANDY
BRIAN 106 101 BELLA
CHEN 201 39 SANDY
RAVI 243 34 SANDY
MAC 288 33 SANDY
```

Note: This sample was generated by hand so there may be an error in it (mental arithmetic error). If so, it will be fixed before the assignment is due.

Implementation Restrictions/ Run-Time/Memory Restrictions

1. An appropriate structure should be used to store customer information.

2. A binary heap structure should be declared and used to implement the priority queues necessary to solve the problem. This structure should internally have **an array of pointers to struct**. The array itself should ALSO be dynamically allocated. (Whenever the heap fills up, the array size storing the heap should be doubled.)

3. Appropriate functions should be written for the binary heap structure.

4. A statically allocated array of (size 10) binary heap structures should be declared in main to store the lines for each stylist.

5. For full credit, your algorithm must run in **O(nlgn)** time by implementing **O(lg n)** operations for the binary heap structure.

6. You must free all dynamically allocated memory for full credit.

7. Note - for arranging the output, you may either output lines as they occur chronologically, or store all of the output lines in a large array, and then sort that array before outputting all of the lines. (There are multiple ways to solve this issue.)

Deliverables

You must submit one file over WebCourses:

1) A source file, *salon.c*. Please use stdin, stdout. There will be an automatic 10% deduction if you read input from a file or write output to a file.

Note: Since this assignment is so challenging, no testing documents will be required to be submitted, since lots of time will be spent coding this assignment.