CSE 2010, HW4

Due Tue Oct 22 at the start of your lab section; Submit Server: class = cse2010, assignment = hw4SxIndividual
Due Tue Oct 22 at the end of your lab section; Submit Server: class = cse2010, assignment = hw4SxGroupHelp
x is 2, 3—your section number (or j for java).

Recent disasters (e.g., hurricanes, earthquakes) and attacks (e.g., shooting) can suddenly increase the number of patients seeking help at emergency rooms. Emergency rooms have plans to prepare for the influx. How would you design a system that can handle the influx efficiently?

The goal of HW4 is to design and implement a system that can efficiently handle the influx of patients at an emergency room. One plan is to triage patients, where patients are prioritized by their level of severity. Some patients might need to be treated immediately, while some others can be delayed.

Consider Emergency Severity Index (ESI) that ranges from 1 (resusitation) to 5 (nonurgent). For simplicity, the number of minutes needed for a doctor to treat a patient is $2^{(6-ESI)}$. Initially, the emergency room has two available doctors (Andrews and Bishop), more doctors may arrive and become available. Furthermore, assume the patient departs after treatment (discharged or admitted to the hospital for further treatment). When two patients have the same ESI, the patient who arrives earlier will be treated first. Time stamps are unique. For this assignment, you can assume at most 200 patients and 50 doctors could be at the emergency room.

To efficiently decide which patient each doctor is going to treat, you will use a priority queue implemented with a heap. Functions/methods include:

- insert(pQueue, entry)
- removeMin(pQueue) // return and remove entry with the minmum key
- getMin(pQueue) // return entry with the minimum key
- isFull(pQueue)
- isEmpty(pQueue)

To implement the priority queue, you may modify/rewrite Programs 9.20 and 9.21 on pp. 352-355 (Java: Code Fragment 9.8 on pp. 377-378 in Goodrich et al.). We will be evaluating your submission on code01.fit.edu; we strongly recommend you to ensure that your submission functions properly on code01.fit.edu.

Input: The command-line argument for hw4.c (HW4.java) is the name of the input file, which has one of the following possible events on each line:

- patientArrives *time patient esi*
- doctorArrives *time doctor*

time is in HHMM format, where HH ranges from 00 to 23 and MM ranges from 00 to 59 (leading zeros are optional). Sample input is on the course website.

Output: Output goes to the standard output (screen):

- patientArrives *time patient esi*
- doctorArrives *time doctor*
- doctorStartsTreatingPatient *time doctor patient*
- doctorFinishesTreatmentAndPatientDeparts *time doctor patient*

Sample output is on the course website.

Extra Credit (10 pts): Separate submission via hw4_extra.c (HW4Extra.java). Consider the condition of a patient can improve or deteriorate over time. Additional possible input action is:

- updatePatientESI *time patient newEsi*

and output result is:

- updatePatientESI *time patient newEsi*

Although the priority queue is designed to find the lowest ESI quickly, it is not designed to find a patient quickly—faster than $O(N)$, where N is the number of patients.

1. Design and implement an additional data structure that can help find a patient and update the priority queue faster than $O(N)$.
2. Note that updatePatientESI might not udpate the entry with the lowest ESI in the priority queue.
3. In the comments at the top of your program (or in a separate PDF file):
   - explain why your additional data structure can help updatePatientESI become faster than $O(N)$ with an analysis of the time complexity of updatePatientESI
   - when updatePatientESI does not remove the entry of lowest ESI, discuss how the heap (priority queue) can be adjusted in $O(\log N)$ time.

[It's OK if patientArrives becomes slower than $O(\log N)$, which can be remedied with data structures to be discussed later in the course.]

Submission: Submit hw4.c (HW4.java) that has the main method, and other program files. Submissions for Individual and GroupHelp have the same guidelines as HW1.

Note the late penalty on the syllabus if you submit after the due date and time as specified at the top of the assignment.