CIS*2520 Assignment Two
Hospital Emergency Room

Due: Oct 23rd at 11:55 pm

Quick Overview

For this assignment, you will simulate the management of patient cases in a minor emergency medical clinic. Your application must use a priority queue. Your ADT priority queue must be implemented using previously created ADT from this course. All ADTs submitted must implement the full API, be your own work, and you may not use third party libraries for this assignment.

Overview

You will write a program that uses a priority queue ADT to simulate the management of an emergency medical clinic. The application will simulate the prioritization and processing of clients as they arrive at the minor emergency clinic. Each simulated client will have a client number, an emergency priority number, and a symptom code. Your application will place the client in priority order in the queue and process the clients through the simulation.

Your simulation must be capable of reading data from a data file in the format specified in this assignment. In addition, patients may be added after reading the initial input. You must create a simple command line interface that supports 3 menu options: 1) Adding a new patient from the command line, 2) Processing the full simulation, and 3) Exiting the program. The user interface must support adding a new patient to the priority queue before processing. From the interface, the user must be able to select any of the three options. Once the simulation is running, it will process all patients and the information will be output to the screen. The program may exit after the user presses any key.

As a bonus task, the priority queue may implement an aging priority starvation algorithm. You must cite sources that describe the known algorithm in your Priority Queue header file. You may alter the algorithm if you wish, but you should describe how you altered it and why.

Your program must not hardcode any part of the input file path. You will submit a tar.gz file to the assignment Courselink Dropbox. The tar.gz file will contain your source code, a make file, your documentation, and your testing files. Your submission must follow the submission guidelines, project structure, and your code must follow common coding standards. Additional, details about the organization of these files are provided later in this document.

Learning Outcomes

When you have finished doing this assignment you should be able to:

- implement and test a Priority Queue ADT

- create a menu interface that activates the simulation program

Required Prior Knowledge

Before you attempt this assignment please ensure that you have a good understanding of the concepts listed below. You can find review material in the course material as well as in the review materials section of the course.

- Linked data structures
- Queues and Priority Queues

Assignment Description

Your simulation must take the name of a data file on the command line (via argv/argc) and use that data to drive the simulation. The data file will consist of one client per line where the data about the client is space delimited. Each line of data will consist of a client ID, a priority, and a symptom code. The client ID will be alphanumeric data with no spaces. All client data will be well-formed but your menu system should handle incorrect input gracefully. The priority will be an integer in the range of 1-5, where 1 is the highest priority and 5 is the lowest priority. The symptom code will be one of the two-character alphanumeric codes shown in the list below. The words and number in parenthesis after the symptom code will not be in the data file, but are provided to help you understand what the symptom codes mean and how many time steps they take to resolve. More information about the symptom codes (presenting complaints) can be found here: http://caep.ca/sites/caep.ca/_les/caep/_les/cedis2008.pdf

- CV (Cardiovascular - 5)
- HN (Ears, mouth, throat, nose 3)
- EV (Environmental - 10)
- GI (Gastrointestinal - 2)
- MH (Mental Health - 5)
- NC (Neurological – 5)
- EC (Ophthalmology -1)
- RC (Respiratory – 1)
- SK (Skin - 2)
- SA (Substance Abuse - 7)
- TR (Trauma - 4)

Use a priority queue to order patients as they arrive at the emergency room. When the user chooses to run the simulation, possibly after inputting more patients via the command line interface, run the simulation and output the results.

Use a counter (a simulation clock) to move the simulation forward 1 "second" at a time. Note that you are not intended to make the simulation run in real time. The simulation proceeds step by step, keeping track of the simulation clock and the time that each patient is processed.

For each patient there is 1 second of paper work that must be completed. The patient's visit will already have ended but the doctor must complete this work before visiting the next patient.

For clarification on any requirements of this assignment, please post in the CourseLink discussion boards.

Input Format

sal45 2 EC

bob52e 1 HN

ted23 5 TR

You may assume that all patients arrive at the same time before the beginning of the simulation.

Output Format

When the simulation finishes your program should output the following report to a file.

- Patient ID
- Priority
- Symptom
- Time finished at the hospital.

It should be formatted as a table to be easily readable and so that the different elements of the report are clearly distinguishable from one another. All information should be displayed in a single row and tabs should be used to separate the data. The information will be output in the order in which is it was input despite the order that it finishes. (Hint) You may want to store patients in two data structures one for processing and another for output.

The output must follow the format shown below.

```
Patient ID          Priority            Symptom Code        Time Finished
Sal45               2                   EC                  6
Bob522              1                   HN                  4
Ted23               5                   TR                  11
```

Submission Guidelines

Your source code must be organized and styled according to the course coding guidelines. Submit a single tar.gz file to the Dropbox. **Do not** use zip, rar, or any other compression program. The tar file should unpack to a folder that is your login name. Inside that folder should be your src/ bin/ include/ assets/ docs/ and lib/ folders. Do not submit any binary files.

Submit all source files, your testing files, a makefile, and a README file.

Follow all other submission guidelines provided for this course.

In your readme file, create a section called starvation algorithm. In this section, describe in 3-5 sentences the starvation algorithm design and how it was applied to the assignment.

Grading

This assignment is worth 10% of your final mark and will be graded out of 100 marks. You can lose marks for not following the submission guidelines, but will not be given marks for that step. Marks are given for successfully demonstrating mastery of the learning outcomes for this assignment.

Marking

The following is not a complete list of all the aspects of an assignment that will be evaluated but provides an overview of the major evaluation components.

/40 Priority Queue ADT

- Priority Queue is an ADT
- Contains and implements all API functions.
- Implements a starvation algorithm
- Manages and frees Memory
- Properly integrates LinkedList API into Priority Queue
- Priority Queue API is properly tested.

/60 Simulation

/10 File Information

- File is passed in via command line arguments.
- File information is correctly read into the program

/40 ER Simulation

- Simulation works as described
- Usability of menu system
  - Checks for input errors
  - Clearly display information
  - Allows patient information to be read and added to simulation
- Priority Queue ADT is properly integrated into the simulation
- Patient and Medical information is logically separated and organized within the project.

/10 Output and Readability

- Information is displayed in requested order and format.

Bonus:

/5 Priority Queue Starvation Algorithm