BITS Pilani
Pilani | Dubai | Goa | Hyderabad

Work Integrated Learning Programmes Division
M.Tech (Data Science and Engineering)

(S1-18_DSEABZG519, S1-18_DSEADZG519, S1-18_DSEAHZG519) –
(Data Structures and Algorithms Design)
First Semester, 2018 -19

Assignment 1 – PS5 - [Hospital Consultation] - [Weightage 10 %]

1. Problem Statement

In this Problem, you have to write an application in JAVA that keeps track of incoming patients in a hospital.

Dr. Kumar runs a hospital for senior citizens which sees a large number of patients coming in everyday. In order to avoid inconvenience to the aged patients, the rule is that the oldest patient is seen first by the doctor. As the patients keep coming and registering, they are added to a priority list from which patient names are taken out for consultation. Since it is not practical to implement this system using pen and paper, an appointment software has to be developed. This software will use Heaps as data structures to keep track of incoming patients and prioritizing them.

The application should be capable to:

1. Take the name of the patient along with his/her age and create a patient ID for the patient.
2. Insert the patient id in the priority list based on the age of the patient.
3. Display the next patient ID in line to meet the doctor. Remove this patient from the priority list.
4. **Perform an analysis of question number 2 and 3 and give the running time in terms of input size,n.**

Data Structures to be used:

PatientRecord: A doubly linked list containing the patient information including the patient name, age and the patient number (assigned by the program).

ConsultQueue: A max heap containing the patient id in the sequence of the next consultation based on the age of the patient.

Functions used by the program:

1. **int registerPatient(String name, int age):** This function registers the name and age of the patient entering the hospital and assigns them an ID that is returned to the calling function. When the program is executed for the first time, the patient details are loaded from an input file. This is analogous to the list of patients present at the hospital before the registration

counter opens. Thereafter, new patients will be input with the help of menu options and console-based input.

2. **void enqueuePatient(Patient_ID):** This function assigns the patient a place in the max heap depending on their age. The patient id is inserted into the max heap. This function should be called every time a new patient is added to the program and should run a sort after adding the patient id to keep the consultation queue updated as per the age condition.
3. **void nextPatient():** This function prints the patient_ID and name of the patient that is next in line to meet the doctor. This function is called either through a menu option of every time a new patient registers and the patient is added to the queue.
4. **void dequeuePatient(Patient_ID):** This function removes the patient ID from the queue that has consulted the doctor and updates the queue. The function is called from the nextPatient function itself after the next patients name is displayed.
5. **void displayQueue():** This function displays all the remaining patients in the queue in the following format: <sequence number> , <patient id>, <patient name>, <age>
   where sequence number is the order in which the patient will meet the doctor. This output should be written in an output.txt file.
6. Include all other functions required to support the operations of these basic functions.

Input file sample

The input may be taken as sequence of ages through a test file – input.txt

Example:

Pradeep, 45

Surya, 60

Ajit, 55

Mary, 64

Radha, 56

2. Deliverables

1. patientNode.java file containing the basic node and associated functions for the patient records
2. patientRecord.java file containing the doubly linked list definition and all operations related to creating the patient record list.
3. consultQueue.java file containing the max heap definition and all operations related to insertion, deletion, sorting and displaying the queue.
4. hospitalConsultation.java containing the main function and menu items to run the program.
5. Input.txt containing the input used to execute the program.
6. Output.txt containing the output of the sorted queue mentioned in operation no. 5

7. A .txt file with answers of question no:4 (Running times).

3. Instructions

- It is compulsory to make use of the data structure/s mentioned in the problem statement.
- It is compulsory to use JAVA for implementation.
- Ensure that all data structure insert and delete operations throw appropriate messages when their capacity is empty or full.
- For the purposes of testing, you may implement some functions to print the data structures or other test data. But all such functions must be commented before submission.
- Make sure that your read, understand, and follow all the instructions

4. Deadline

- The strict deadline for submission of the assignment is **Jan 5, 2019 eod.**
- Late submissions won't be evaluated.

5. How to submit

- This is a group assignment.
- Each group consists of 8-11 members from each batch. All members of the group will work on the same problem statement.

  Ex: DSAD_[BLR/HYD/DLH]_GROUP[1-5]

  The group details can be found here.
- Each group is further divided into subgroups.

  Ex: [BLR/HYD/DLH]_[B1-B4]_[G1-G5]_SUBGROUP [1]
- Each subgroup consists of 2-4 members.
- If your group has 11 students, there will be 3 subgroups of 3 students each and 1subgroup with 2 students. If your group has 10 students, there will be 2 subgroups of 3 students and 1 subgroup of 4 students.
- Each subgroup has to make one submission (only one, no resubmission) of solutions.
- Each sub group should zip the deliverables into "ASSIGNMENT1_[BLR/HYD/DLH]_[B1-B4]_[G1-G5]_SUBGROUP [1-4].zip" and upload in CANVAS in respective location under ASSIGNMENT Tab.
- Assignment submitted via means other than through CANVAS will not be graded.

6. Evaluation

- The assignment carries 10 Marks
- Grading will depend on
  - Fully executable code with all functionality

    - Well-structured and commented code

- Every bug in the functionality will cost you 0.5 mark each.
- Every segmentation fault/memory violation will cost you 1 mark each.
- Source code files which contain compilation errors will get at most 25% of the value of that question.

7. Readings

- **Section 2.4,2.6:** Algorithms Design: Foundations, Analysis and Internet Examples Michael T. Goodrich, Roberto Tamassia, 2006, Wiley (Students Edition)