Data Structures and Algorithms

COMP 103 2019-20 Semester 2

Lecture 09b

Dr. Kerese Manueli

kerese.manueli@ecs.vuw.ac.nz

Victoria University of Wellington

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core:

Complete the `HospitalERCore` and `Patient` classes so that it simulates patients coming to the emergency room, and reports on basic statistics.

1. Complete the `reset` method to set up `waitingRoom` to have a queue of patients and `treatmentRoom` to have a set of Patients.
2. Complete the run method. Each loop of the `run()` method represents one tick of time. The method should:
   - ✓ Advance the time by one "tick"
   - ✓ Find all patients in the treatment room who have completed their current treatment and discharge them (remove them from the treatment room). For debugging, it is helpful to print a message for each patient when they are discharged: `UI.println(time+ ": Discharge: " + p);`
   - Process one time tick for each patient currently being treated, or waiting in the waiting room.
   - Move patients from the waiting room to the treatment room if there is space.
   - Get any new patient that has arrived and adds them to the waiting room (done for you already).
   - Display the state of the waiting room and treatment room (done for you already).

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

2. Complete the run method. Each loop of the `run()` method represents one tick of time. The method should:

   - Advance the time by one "tick"
   - ✓ Find all patients in the treatment room who have completed their current treatment and discharge them (remove them from the treatment room).
   - ✓ For debugging, it is helpful to print a message for each patient when they are discharged: `UI.println(time+ ": Discharge: " + p);`
   - Process one time tick for each patient currently being treated, or waiting in the waiting room.

```
37  * Methods:
38  *   compareTo(other)    -> int
39  *   redraw(x, y)
40  *   toString()          -> String
41  *   getPriority()       -> int
42  *   getWaitingTime()    -> int
43  *   getTreatmentTime() -> int
44  *
45  *   waitForATick()
46  *   advanceTreatmentByTick()
47  *   completedCurrentTreatment() -> boolean
48  *
49  *   [THE FOLLOWING ARE NOT NEEDED FOR THE CORE]
50  *   noMoreTreatments()    -> boolean
51  *   incrementTreatmentNumber()
52  *   getCurrentTreatment()       -> String (name of department for the treatment needed now)
53  */
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

2. Complete the run method. Each loop of the `run()` method represents one tick of time. The method should:

   - Advance the time by one "tick"
   - ✓ Find all patients in the treatment room who have completed their current treatment and discharge them (remove them from the treatment room).
   - ✓ For debugging, it is helpful to print a message for each patient when they are discharged: `UI.println(time+ ": Discharge: " + p);`
   - Process one time tick for each patient currently being treated, or waiting in the waiting room.

```
127         List<Patient> temp = new ArrayList<Patient>(); // Create Temporary List for patients
128         for (Patient p : treatmentRoom) {              // Loop to check patients in treatmentRoom
129             if (p.completedCurrentTreatment()) {       // if patient has completedCurrenTreatment
130                 temp.add(p);                           // then add to Temporary List
131             }
132         }
133         for (Patient p : temp) {
134             treatmentRoom.remove(p);
135             UI.println(time+ ": Discharge: "+p);
136         }
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

2. Complete the run method. Each loop of the `run()` method represents one tick of time. The method should:

   - Advance the time by one "tick"
   - ✓ Find all patients in the treatment room who have completed their current treatment and discharge them (remove them from the treatment room).
   - ✓ For debugging, it is helpful to print a message for each patient when they are discharged: `UI.println(time+ ": Discharge: " + p);`
   - Process one time tick for each patient currently being treated, or waiting in the waiting room.

```
127     List<Patient> temp = new ArrayList<Patient>(); // Create Temporary List for patients
128     for (Patient p : treatmentRoom) {              // Loop to check patients in treatmentRoom
129         if (p.completedCurrentTreatment()) {       // if patient has completedCurrenTreatment
130             temp.add(p);                           // then add to Temporary List
131         }
132     }
133     for (Patient p : temp) {
134         treatmentRoom.remove(p);
135         UI.println(time+ ": Discharge: "+p);
136     }
```

Type these lines of code after code line 136

```
for (Patient p : treatmentRoom) {
    p.advanceTreatmentByTick();
}
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

2. Complete the run method. Each loop of the `run()` method represents one tick of time. The method should:

   - Advance the time by one "tick"
   - ✓ Find all patients in the treatment room who have completed their current treatment and discharge them (remove them from the treatment room).
   - ✓ For debugging, it is helpful to print a message for each patient when they are discharged: `UI.println(time+ ": Discharge: " + p);`
   - Process one time tick for each patient currently being treated, or waiting in the waiting room.

```
37  * Methods:
38  *   compareTo(other)   -> int
39  *   redraw(x, y)
40  *   toString()         -> String
41  *   getPriority()      -> int
42  *   getWaitingTime()   -> int
43  *   getTreatmentTime() -> int
44  *
45  *   waitForATick()
46  *   advanceTreatmentByTick()
47  *   completedCurrentTreatment() -> boolean
48  *
49  *   [THE FOLLOWING ARE NOT NEEDED FOR THE CORE]
50  *   noMoreTreatments()    -> boolean
51  *   incrementTreatmentNumber()
52  *   getCurrentTreatment()       -> String (name of department for the treatment needed now)
53  */
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

2. Complete the run method. Each loop of the `run()` method represents one tick of time. The method should:

   - Advance the time by one "tick"
   - ✓ Find all patients in the treatment room who have completed their current treatment and discharge them (remove them from the treatment room).
   - ✓ For debugging, it is helpful to print a message for each patient when they are discharged: `UI.println(time+ ": Discharge: " + p);`
   - Process one time tick for each patient currently being treated, or waiting in the waiting room.

```
127         List<Patient> temp = new ArrayList<Patient>(); // Create Temporary List for patients
128         for (Patient p : treatmentRoom) {              // Loop to check patients in treatmentRoom
129             if (p.completedCurrentTreatment()) {       // if patient has completedCurrenTreatment
130                 temp.add(p);                           // then add to Temporary List
131             }
132         }
133         for (Patient p : temp) {
134             treatmentRoom.remove(p);
135             UI.println(time+ ": Discharge: "+p);
136         }
```

```
for (Patient p : treatmentRoom) {
    p.advanceTreatmentByTick();
}
```

Type these lines of code after } sign

```
for (Patient p : waitingRoom){
    p.waitForATick();
}
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core:

Complete the `HospitalERCore` and `Patient` classes so that it simulates patients coming to the emergency room, and reports on basic statistics.

1. Complete the `reset` method to set up `waitingRoom` to have a queue of patients and `treatmentRoom` to have a set of Patients.
2. Complete the run method. Each loop of the `run()` method represents one tick of time. The method should:
   - ✓ Advance the time by one "tick"
   - ✓ Find all patients in the treatment room who have completed their current treatment and discharge them (remove them from the treatment room). For debugging, it is helpful to print a message for each patient when they are discharged: `UI.println(time+ ": Discharge: " + p);`
   - ✓ Process one time tick for each patient currently being treated, or waiting in the waiting room.
   - Move patients from the waiting room to the treatment room if there is space.

```
if (!waitingRoom.isEmpty() && treatmentRoom.size() < MAX_PATIENTS){
    treatmentRoom.add(waitingRoom.poll());
}
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core:

Complete the `HospitalERCore` and `Patient` classes so that it simulates patients coming to the emergency room, and reports on basic statistics.

1. Complete the `reset` method to set up `waitingRoom` to have a queue of patients and `treatmentRoom` to have a set of Patients.
2. Complete the run method. Each loop of the `run()` method represents one tick of time. The method should:
   - ✓ Advance the time by one "tick"
   - ✓ Find all patients in the treatment room who have completed their current treatment and discharge them (remove them from the treatment room). For debugging, it is helpful to print a message for each patient when they are discharged: `UI.println(time+ ": Discharge: " + p);`
   - ✓ Process one time tick for each patient currently being treated, or waiting in the waiting room.
   - ✓ Move patients from the waiting room to the treatment room if there is space.
   - Get any new patient that has arrived and adds them to the waiting room (done for you already).
   - Display the state of the waiting room and treatment room (done for you already).

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

3. Add fields to record basic statistics, extend the `run()` method to update the statistics when a patient is discharged, and complete the `reportStatistics()` method by calling the `reportLine()` method to report:
   - The total number of patients treated
   - The average waiting time of the patients

```
147             // Gets any new patient that has arrived and adds them to the waiting room
148             if (time==1 || Math.random()<1.0/arrivalInterval){
149                 Patient newPatient = new Patient(time, randomPriority());
150                 UI.println(time+ ": Arrived: "+newPatient);
151                 waitingRoom.offer(newPatient);
152             }
153             redraw();
154             UI.sleep(delay);
155         }
156         // paused, so report current statistics
157         reportStatistics();
158     }
159
160     // Additional methods used by run() (You can define more of your own)
161
162     /**
163      * Report summary statistics about all the patients that have been discharged.
164      * (Doesn't include information about the patients currently waiting or being treated)
165      * The run method should have been recording various statistics during the simulation.
166      */
167     public void reportStatistics(){
168         /*# YOUR CODE HERE */
169
170     }
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

3. Add fields to record basic statistics, extend the `run()` method to update the statistics when a patient is discharged, and complete the `reportStatistics()` method by calling the `reportLine()` method to report:
   - The total number of patients treated
   - The average waiting time of the patients

```
162     /**
163      * Report summary statistics about all the patients that have been discharged.
164      * (Doesn't include information about the patients currently waiting or being treated)
165      * The run method should have been recording various statistics during the simulation.
166      */
167     public void reportStatistics(){
168         /*# YOUR CODE HERE */
169         reportLine();
170     }
171
172     public void reportLine(){
173         /*# YOUR CODE HERE */
174         UI.println("Total number of patients is "+numberOfPatients);
175         UI.println("Average waiting time of patients is "); //
176     }
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

4. Make the waiting room act as a priority queue taking into account the priority of the patients.
   - Extend the `reset` method to use a priority queue if the argument to the method is true, and a regular queue if the argument is false.

```
93      public void reset(boolean usePriorityQueue){
94          running=false;
95          UI.sleep(2*delay);  // to make sure that any running simulation has stopped
96          time = 0;           // set the "tick" to zero.
97
98          // reset the waiting room, the treatment room, and the statistics.
99          /*# YOUR CODE HERE */
100
101         waitingRoom = new ArrayDeque<Patient>();
102         treatmentRoom = new HashSet<Patient>();
103
```

```
if (usePriorityQueue){
    waitingRoom = new PriorityQueue<Patient>();
}
else {
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

4. Make the waiting room act as a priority queue taking into account the priority of the patients.
   - Extend the `reset` method to use a priority queue if the argument to the method is true, and a regular queue if the argument is false.
   - Complete the `compareTo` method in the `Patient` class so that a Patient with a higher priority is ordered before a Patient with lower priority. Patients with equal priority should be ordered by their time of arrival.

```
154     /**
155      * Compare this Patient with another Patient to determine who should
156      *  be treated first.
157      * A patient should be earlier in the ordering if they should be treated first.
158      * The ordering depends on the triage priority and the arrival time.
159      */
160     public int compareTo(Patient other){
161         /*# YOUR CODE HERE */
162
163         return 0;
164     }
```

```
if (this.priority < other.priority) { return -1; }
if (this.priority > other.priority) { return 1; }
```

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

4. Make the waiting room act as a priority queue taking into account the priority of the patients.

   - Extend the `reset` method to use a priority queue if the argument to the method is true, and a regular queue if the argument is false.
   - Complete the `compareTo` method in the `Patient` class so that a Patient with a higher priority is ordered before a Patient with lower priority. Patients with equal priority should be ordered by their time of arrival.
   - Add fields, and modify the `run()` and `reportStatistics` methods to also report
     - The total number of priority 1 patients treated,
     - The average waiting time of the priority 1 patients

     Hint: You need to call `reportLine()` method twice. Once for all patients, and once for just patients with priority 1.

Run some tests to see whether using the priority queue improves the waiting time for priority 1 patients.

```
93      public void reset(boolean usePriorityQueue){
94          running=false;
95          UI.sleep(2*delay);  // to make sure that any running simulation has stopped
96          time = 0;           // set the "tick" to zero.
97
98          // reset the waiting room, the treatment room, and the statistics.
99          /*# YOUR CODE HERE */
100
101         waitingRoom = new ArrayDeque<Patient>();
102         treatmentRoom = new HashSet<Patient>();
103
104         totalWaitingTime = 0;
105         numberOfPatients = 0;
```

Need to add fields for priority 1 patients

Assignment 3a

https://ecs.wgtn.ac.nz/Courses/XMUT103_2020T1/Assignment3PartA

Core

4. Make the waiting room act as a priority queue taking into account the priority of the patients.
   - Extend the `reset` method to use a priority queue if the argument to the method is true, and a regular queue if the argument is false.
   - Complete the `compareTo` method in the `Patient` class so that a Patient with a higher priority is ordered before a Patient with lower priority. Patients with equal priority should be ordered by their time of arrival.
   - Add fields, and modify the `run()` and `reportStatistics` methods to also report
     - The total number of priority 1 patients treated,
     - The average waiting time of the priority 1 patients

       Hint: You need to call `reportLine()` method twice. Once for all patients, and once for just patients with priority 1.

Run some tests to see whether using the priority queue improves the waiting time for priority 1 patients.

```
177     public void reportLine(int i){
178         /*# YOUR CODE HERE */
179
180         if (i==0) {
181             UI.println("Total number of patients is "+numberOfPatients);
182             UI.println("Average waiting time of patients is ");
183         }
184         else {
185             UI.println("Total number of patients is "+numberOfP1Patients);
186             UI.println("Average waiting time of Prioty 1 patients is ");
187         }
188     }
```