Daily Surgery Room Scheduling in METU Hospital

The head doctor of a local hospital wants to schedule the medical operations for daily surgery rooms. The doctor makes the schedules for the following 5 consecutive days. Operations that are performed for each day will be determined and scheduled according to some constraints, explained below:

- There are certain number of surgery rooms in the hospital, and the operations should be placed only in one of these rooms. In each room only one operation will be performed at a given time. The number of rooms is fixed and provided initially.
- You may assume that the number of doctors is sufficient to serve the patients.
- The time horizon at which you will perform scheduling is between 9:00 and 17:00 for each day. You may take this interval as [0, 480]. The data provided for each day is given as being in this interval.
- Each operation has a fixed operation time and it needs to be completed within the given time window.
- The operations have to take place in the given time interval. For instance, if the time window for an operation is [0,30], and its duration is 10 minutes, then it cannot start at the 21st minute as the latest start time, since the end of the operation exceeds the time interval.
- There is a priority level for each patient who will take the operation, and it is represented

  by a 4-level integer value as 1, 2, 3 and 4. While Level-1 denotes the highest priority, Level- 4 denotes the least priority.
- Priority mainly refers to the waiting time for an operation. The patient with the higher priority should spend less waiting time in the system than the patients having lower priorities.
- Suppose you have two patients whose operations need to be completed in the time window [0,20] and each takes 10 minutes to operate. Then the one with higher priority should be completed before the other.
- The operations in the same room cannot overlap.
- As the duration of the operations is fixed, you can assume that there is no change on the operation duration. (so the operation duration is not probabilistic)
- When an operation cannot be scheduled in its original day, you need to make sure that the corresponding operation is scheduled in the very next day. This operation cannot be postponed two days in a row. Therefore, you need to apply the following steps:
  - Update the priority level to Level- 0, representing the emergency level and it is a higher priority than the Level- 1. This priority level is ONLY given to the cases which are postponed to the next day.

  - Start time of the time window for the operation should be updated to 9:00 and the end time of the time window should be set as the summation of the start time and the duration of the operation.
  - The day of the patient will be updated.

- The scheduling algorithm should place all the operations in the list according to the rules above.

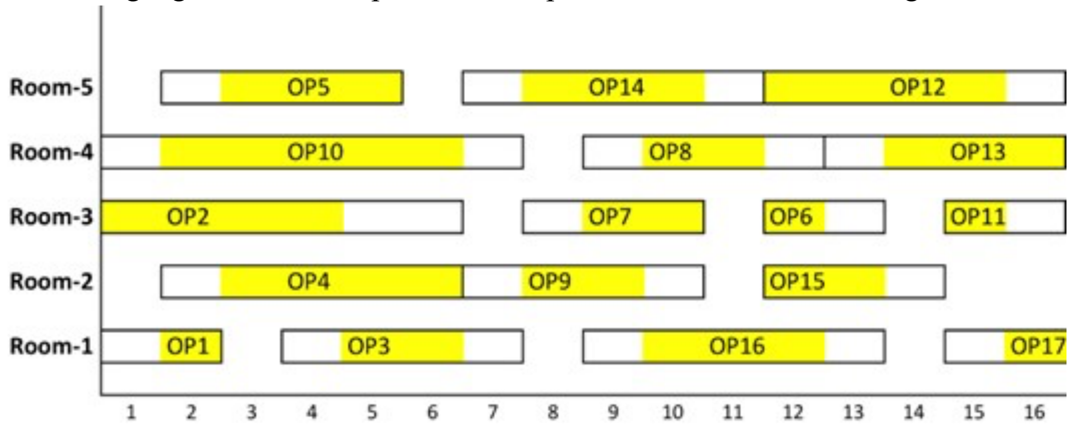

Figure 1: An example of scheduled operations with presenting the given time windows and operation durations

The framed boxes in Figure 1 indicate the time intervals during which the operations can be scheduled and the actual schedules of the operations in that time intervals are denoted with yellow color for each room for a given day.

We provide the design of the classes that you will be using in this project in Figure 2. The diagrams will be helpful when you later write your code that involves interactions among the objects of the different classes. Please read carefully the details of the classes and objects below.

Mainly, the following terminology will be used in the project: *operation*, *schedule*, *patient* and *interval*. These are the objects that we will use. An *Interval* has a left (start) and right (end)

hand. An *Operation* has an ID, patient, duration, a time window (available interval) during which the surgery can take place, scheduled interval during which the surgery is planned to be performed, and the day that the operation is performed. *Patient* has a name, surname, priority level which is between 1 and 4 describing her/his situation, and the day that the patient is initially listed to have the operation. A *Schedule* contains daily planning horizon, total planning days that the schedule will be performed, number of available rooms in the hospital, and final schedule of the operations.

Our design involves four classes: *Interval*, *Operation*, *Schedule* and *Patient*. We then write a main script to read event data from an Excel file and then instantiate objects of the different classes in order to create a feasible schedule.

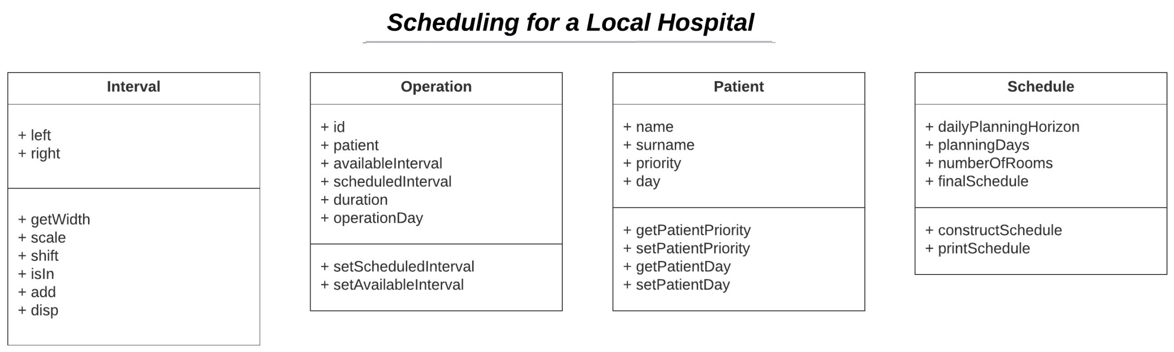

Figure 2: Diagram for the Classes

So which class should you work on first? It would be better to start with the most independent class, the one that does not depend on other classes. So start with class *Interval*. As you complete the classes, do not change the names of the properties, methods and parameters. You may add extra methods or properties to the classes if needed (DO NOT define additional classes.), but make sure that you explain your additional declarations in detail, both in the report and in the script as a comment.

1) **Interval** Class

Class Interval has left and right properties. Description of these properties is as follows

- left: start point of the interval
- right: endpoint of the interval

Description of the methods is as follows:

- getWidth: Gives the length of the interval.
- scale: Scales the interval by a factor.
- shift: Shifts the interval by a constant.
- isIn: Checks whether a given interval is included in another interval. It returns true (1) if it is included, false otherwise.
- add: Adds predetermined values to the start and endpoints to the interval (the values that will be added to start and endpoints may differ).
- disp: Displays the interval in this format: (left,right).

2) Operation Class

An operation has an ID, a Patient, available Interval (time window) in which it can be scheduled, a duration, and a time at which it’s finally scheduled. Description of the properties is as follows:

- id: Id number of the operation
- patient: Patient (object of type Patient)
- availableInterval: Available interval for scheduling an operation (object of type Interval)
- scheduledInterval: Scheduled start time (object of type Interval)
- duration: Length of the operation in minutes
- operationDay: the day that the operation will be held (Initially no value is assigned to this property. It will take a value when the operation is scheduled.)

Description of the methods are as follows:

- setScheduledInterval: A setter method to set scheduledInterval property of the object Operation when it is scheduled.
- setAvailableInterval: A setter method to set availableInterval property of the object Operation when the operation is to be postponed to the next day.

3) Patient Class
A patient has a name, surname, priority level, and a day. Description of the properties is as follows:

a. name: Name of the patient

b. surname: Surname of the patient

c. priority: Priority of the patient’s situation

d. day: the day that the patient initially listed to have an operation

Description of the method is as follows:

e. getPatientPriority: A getter method to obtain the priority level of the object Patient.

f. setPatientPriority: A setter method to set the priority level of the object Patient.

g. getPatientDay: A getter method to obtain the day of the object Patient.

h. setPatientDay: A setter method to set the day of the object Patient.

4) Schedule Class
A schedule has a daily planning horizon, total planning days number of rooms, and a final schedule. Description of the properties is as follows:

- dailyPlanningHorizon: Planning horizon in a day that the event can be scheduled (object of type Interval)
- planningDays: total number of days that the schedule will be planned
- numberOfRooms: Number of rooms on hand

- finalSchedule: The cell array of scheduled operations (Initially no value is assigned to this property. This property will take a value after the schedule has generated)

Description of the methods is as follows:

- constructSchedule: Adds operations to the property finalSchedule with the help of a heuristic that you will construct. It is not the constructor method of the class. Therefore, in the beginning the property finalSchedule is an empty cell array, and it will be updated in this method.
- printSchedule: Prints the resulting schedule in a proper format.

Your Task

You need to design heuristics for scheduling the operations for the given objectives. In other words, you need to construct two heuristics to solve the scheduling in order to serve for each objective function, see below.

- schedule objective 1: The first objective is to maximize the number of operations which is completed within its initial available time interval.
- schedule objective 2: The second objective is to maximize the utilization of the rooms.

How to run?

You will need a main script to run your program. In this script, you will be reading the data from an Excel file, *InputData.xslx*, instantiate the objects that you need by the help of classes that you have defined. Then with the help of these objects, you will form the final schedule using the method *constructSchedule* in the Schedule class. Then, you need to print your schedule with the help of *printSchedule* method in the Schedule class. In other words, you will use main script for only reading data, instantiating the objects, calling the required methods of classes, and printing the final schedule.

Output Format

Suppose you have two rooms in the hospital. *printSchedule* method, included in class Schedule, should give an output as follows:

Table 1: Sample Output

Room No	**Available Interval**	**Duration (min)**	**Sched. Interval**	**Patient Name**	**Patient Surname**	**Patient Priority**	**Operat-ion Day**
1	(0,40)	20	(0,20)	Dilay	Ozkan	1	1
1	(0,40)	20	(20,40)	Alper	Sener	2	1
1	(40,120)	60	(40,100)	Melissa	Karagur	3	1
1	(240,480)	120	(240,360)	Ceyhan	Sahin	1	2
2	(60,160)	100	(60,160)	Can	Er	1	2
2	(20,80)	40	(20,60)	Beril	Akkaya	2	2

Please note that the schedule given above is only an example. According to your objective, it may change. You will give the output,

- in an Excel file,
- as a Gantt Chart (for each day).

After explaining the two algorithms developed, you are expected to analyze the following issues as well:

- For the *schedule objective 1* report the following:
  - the objective function value,
  - utilization of the rooms,
  - the number of operations shifted in each priority levels.
- For the *schedule objective 2* report the following:
  - the objective function value,
  - the number of operations that are postponed to the next day,
  - the number of operations shifted in each priority levels.

  Moreover, answer the following questions. Support your answers using your computational environment.

- What if the number of rooms is increased by 1 for both objectives? What changes in the schedules?
- Considering the new case above, is the usage of rooms fair or is there a room used more than the others?