The American University in Cairo
Computer Science and Engineering Department

Fundamentals of Computing II
F2020
CSCE110102/03
Assignment 2
Object Oriented Programming fundamentals

Create a small hospital system, where patients can book appointments. The program should be able to take a patient's information and the appointment that they'd like to book. It should also be able to display the patients names in order of their appointments (ascendingly).

Your program should consist of :
<u>Class Person</u> : with private variables name, ID and age. A default and a copy constructor, setters and getters and a print function.

<u>Class Patient</u> : inherits from Person.

Private variables:
struct appointment : consisting of hours and mins. (*You can assume we're using the 24 hours system*)

doctorID: to indicate the ID of the doctor assigned to their case. (*can be any number*)

Overload the <,>,== operators to compare between patients' appointments.

Add any setters/getters necessary.

<u>Class Doctor</u>: inherits from Person.

Private variables :
counter : number of appointments the Dr has for the day
Array of struct (appointment): that indicates the times the Dr is booked.

A function that returns true if the Dr is available at a certain time, false if not.

Add any setters/getters necessary

<u>Template Class Queue</u>

A generic queue class to work with any data type.
Main functions : Push, Pop.

Program Workflow :

- You could read multiple Doctor's data from the user/from a file or initialize them within the main function.
- Read multiple patients data from a file or from the user. Insert patients with their appointment into an array of patients.
- The program should assign each patient to the doctors in the order the doctors are stored so the first patient would go to the first doctor, and the second patient to the second doctor and so on till you go back to the first doctor.
- If a doctor is not available in the appointment the patient chose then the next doctor should be considered and so on. (Use the doctor class functions to check the doctor's availability). *You can assume you'll always find an available doctor for each patient.*
- The patients should be inserted into a queue in order of their appointments.
- Print the information of the patients in order, along with the info of their assigned doctor.

Construct a UML to represent the three classes and the relationships between them.

Please make sure to document your code by adding comments that explain your logic and the functions you're creating whenever possible.

Test Case:

Patients:
Ahmed at 1:00, Sara at 4:00, and Kareem at 3:00, Mohammed 1:00

Doctors :
Ayman, Khaled, Mai

Ahmed should be assigned to Dr Ayman 1:00
Sara should be assigned to Dr Khaled at 4:00
Kareem should be assigned to Dr Mai at 3:00
Mohammed should be assigned to Dr Khaled at 1:00 (since Dr Ayman is unavailable at 1:00 and Dr Khaled is next in the array)

After inserting the patients to the queue, they should be in the following order.

Ahmed has an appointment at: 1:00. with Dr Ayman
Mohammed has an appointment at 1:00 with Dr Khaled
Kareem has an appointment at 3:00 with Dr Mai
Sara has an appointment at: 4 : 0. with Dr Khaled

Hints:

To read patients/doctors from a file easily, make sure to use the same format for each Dr/Patient in the text file.

We read the file line by line and save the data in the corresponding variables in the program. So each time we run the program the array of patients/doctors can be already filled with data (In which case we don't take them from the user)

Example for the doctors.txt:

```
Mai // Dr name
40  // Dr age
1 // Dr ID
3 // number of appointments, should be followed by 3*2 lines that indicate the hour & mins of
each appointment.
1      // appointment 1 at 1:00
0
5      // appointment 1 at 5:30
30
3     // appointment 3 at 3:15
15
```

Then we ***for each*** doctor in the array of doctors :

```
doctor x;
x.name= ((The first line which is Dr name))
x.age= ((The second line which is Dr age))
x.ID= ((The third line which is doctor ID ))
x.num = ((The fourth line which is the number of appointments))
```

Then loop according to the number of appointments to read the time of each appointment line by line into the x.appointments array. So the hours for appointment 1, for example, would be stored in x.appointments[0].hours, and the minutes for appointment 1 would be stored in x.appointments[0].mins.