SOFT_7004 Project OO Principles

Completion Date: 11th December (End of week 11) 50 % of module marks.

Hospital Consultancy Application

A group of Consultant (s) want to manage their practice. The practice has four soon to be five or more consultants. Consultants have patients. Patients meet with consultants.

Write an application to manage Consultants, patients and patient's visits with consultants. We are recording patient visits that have occurred not an appointment system.

Each class written should have getters and setters for each field and a toString and a .equals method.

Java Classes Required

1. A **Name** class. This stores details of a name
   ```
   String firstname
   String lastName
   ```

2. A **Person** class. This is a super class for all people in the Application. Its attributes are:

   ```
   Name name
   String id
   String phone
   ```

3. A **Practice** class. This holds a collection of Consultants.

   ```
   List of Consultants

   Operations:
   Add Consultant
   show consultant list
   find a consultant
   Show details for all consultants
   Find a Paitent
   ```

4. A **Consultant** class. This is a subclass of Person. Consultant's have a collection of patients and an expertise

   ```
   List of Patients
   String expertise

   Operations:
   Add patients
   ```

```
Add a patient visit
Show all Patients
Show all Patients and Visits
```

5. A **Patient** class. This class is a subclass of Person. It also holds an illness description and a severity. Severity is graded one to five, one is mild, five is severe. Each patient has a collection of visits.

```
List of Visits
String illness
enum severity
```

Operations:

```
Add a visit
show all visits
```

6. A **Visit** class. This records a visit by a patient to the consultant that they are attending. This will have a the following fields:

```
Date  date of a visit
String notes
```

Design

You are to create a UML diagram which details all the java classes used in your application and the relationship between them.

Test Code

As you write each class you should implement tests for that class in your test class. Once all the classes are written you should have a test class which (via hard coding)

- creates a practice
- adds 3/4 consultants
- add patients consultants
    - Consultants should have have 0 1 or 1+ patients
- adds visits for patients.
    - Patients should have have 0 1 or 1+ visits
- display all consultants followed by a list of patients – showing the visits for those patients.

Application Code

Provide a menu (command line driven) in your Test class which has the following options:

1. Record a Patient visit.
2. Display all patients of a particular Consultant.
3. Display all Consultants followed by their Patients followed by their visits.
4. Load information from a text file. (see below)
5. Quit

The following is a sample of how the data may be stored. This should be loaded by option 4 in the menu.

- Patient 1223 and 7213 are attending consultant Harry Walsh.
- Patient 1223 has attended for a visit twice. Patient 7213 has had 1 visit.
- Patient 3412 is attending consultant Adam Walsh and has had no visit yet.

Practice.txt

```
Consultant,Walsh,Harry,812,hw@x.com,Cardiology
Patient,Murphy,Annie,1223,hw@x.com,Angina,4
Visit,01/12/2020,Initial Consult
Visit,01/12/2020,Pacemaker
Patient,Murphy,Danny,7213,hw@x.com,Arithmyia,2
Visit,01/12/2020,ECG work
Consultant,Walsh,Adam,912,hw@x.com,Cancer
Patient,Murphy,Tommy,3412,hw@x.com,Skin,4
```

Note: the order of the line in the file is significant.

Marking Scheme

Coding Readability + Design	10%
Practice with Consultants Add , show , find	20%
Patients and Visits (Add and show)	20%
Hard coded testing	15%
Menu Application Option 1,2, 3	20%
Menu Option 4 load data	15%