Dictionary Project

Problem

Write a program that works as a patient data base for a hospital or doctor's office.
Your implementation should have three classes:

- Visit
- PatientRecord
- PatientDataBase
- PatientDataBaseDriver

Visit.java

The Visit class encapsultes a visit to the hospital or doctor. It should contain data for the date of the visit, the reason for the visit, and the treatment given during the visit. Reason and Treatment should be strings while Date can be a string or a LocalDate. Your class should have a constructor that takes values for the three data fields as well as getters. You can also have setters if you would like, but it makes more sense to make this an immutable class with only getters. You should override equals and hashCode since you will need to be able to determine if two visits are the same in the PatientRecord class. You should also override the toString method.

PatientRecord.java

The PatientRecord class should have data for the patient's first and last name, data of birth, and a list of visits. You should have at least one contructor that takes the patient's first and last name and date of birth. You may additionally have a constructor that takes the patient's first and last name, date of birth, and a list of visits. First and last name should be strings. Date of birth may be a string or a LocalDate. List of visits should be a list such as an ArrayList.

The class should have a getter for each of the four data fields. You may also have setters, but you should consider making your fields immutable. When you code your getter for visits, make sure the client cannot modify your visit list from outside your class.

You should be able to add a visit and remove a visit from the class. You will need to override the equals and hashCode methods. The PatientDataBase class will use the hashCode method for mapping. You should also overridet the toString() method.

Hash Used in PatientRecord

For the PatientRecord class, I used the following hash:

```
@Override
public int hashCode() {
   int hash = 5;
   hash = 79 * hash + Objects.hashCode(this.firstName);
   hash = 79 * hash + Objects.hashCode(this.lastName);
   hash = 79 * hash + Objects.hashCode(this.DOB);
   return hash & 0x7fffffff;
}
```

The last line causes the hash to always be a positive number which makes the patient id more reasonable.

PatientDataBase.java

The PatientDataBase class should maintain a list of PatientRecords in a hash table. You are free to use HashMap or HashSet found in the Java API. For the key, you should use the same hash that is calculated in the PatientRecord class. The value should be the PatientRecord that corresponds to that hash.

You should have at least one constructor that takes a string for a file name. This constructor is responsible loading the patient records from the file into the hash table.

You should have a method that writes out the patient database to a text file named with the parameter value passed into the method. The format should be the same as you read in.

You should have methods to perform the following queries:

- Given a patient record, add the patient record
- Given the first and last name and date of birth, add the patient record
- Given a patient record, remove the patient record
- Given the first and last name and date of birth, remove the patient record
- Given the patient id (patient record hash code), get the patient record
- Given the first and last and DOB, return the patient record
- Get a list of patient names (last name, first name) and the patient id, sort by patient last name. See https://www.geeksforgeeks.org/sorting-a-hashmap-according-to-values/ for help on this.
- Given the patient id (patient record hash code) and date, return the reason for the visit; if there is more than one visit on a date, return all the reasons
- Given the first and last and DOB and date, return the reason for the visit; if there is more than on visit on a date, return all the reasons
- Given the patient id (patient record hash code) and date, return the treatment for the visit; if there is more than one visit on a date, return all the treatments
- Given the first and last and DOB and date, return the treatment for the visit; if there is more than one visit on a date, return all the treatments
- Given the patient id (patient record hash code), return a list of dates of all visits
- Given the first and last name and DOB, return a list of dates of all visits
- Given the patient id (patient record hash code), return the paired reason/treatment for the visit; if there is more than one visit on a date, return all paried reasons/treatments
- Given the first and last name and DOB, return the paired reason/treatment for the visit; if there is more than one visit on a date, return all paried reasons/treatments
- Given the patient id (patient record hash code) and a Visit, add the visit to a patient record
- Given the first and last name, DOB and a Visit, add the visit to a patient record

PatientDataBaseDriver.java

This class has been provided for you. It should present the user with a menu with options that will test the patient database files. You may write your own PatientDataBaseDriver class or use the one provided.

Pair.java

This is a helper class that is used in PatientDataBase and PatientDataBaseDriver classes. This class has been provided for you and you are welcomed to use it if you would like.

Patient Records File Format

A sample input file has been included with this assignment.
The format of the file is as follows:

```
? This is a comment
? Record number, this is not the same as patient id
? First name
? Last name
? DOB
? Date of visit
? Reason for visit
? Treatment provided at visit
1
Jack
Sprat
1998-07-22
2018-02-18
Cannot eat fat
Placed on fat free diet
2
Miss
Muffet
2000-04-05
2014-03-07
Stomach ache from eating curds and whey
Given Pepto Bismol
2015-05-25
Irrational fear of spiders
Given anti-spider serum
```

Any line starting with a question mark (?) is a comment and should be ignored. The first line of actual data is a record number. This is not the same as the patient id number. The next three lines are patient first name, patient last name, patient DOB. Patient visits are listed next in groups of three lines. Date of visit, reason for visit, treatment provided at visit. Visit lines will repeat for each visit by the patient. The start of a new record is indicated by a new record number.

Input

The input is initially a file of patient records. It should also be possible for the user to insert new patient records or remove patient records. The initial input file should be updated to reflect insertion and deletion of records.

Output

Your output should be to the console. In addtion, if the user makes any changes to the database, the user should have the option of saving the changes to a file.

Requirements

You may write your code in your language of choice provided I know the language. You should use a dictionary when writing your PatientDataBase class. The dictionary can be a map from the Java API or C++ STL.

What to submit

Create a folder called DictionaryProject_lastName_firstName replacing lastName with your actual last name and firstName with your actual first name. Inside the folder place copies of all source files that you wrote for this project. You should compress (zip) the folder and then submit the compressed folder to Blackboard.

If you wrote your project in another languge besides java please include a readme file explaining how to run your code. Make sure all code is well-documented.

Sample Output

```
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 1
****************************

*************************************
1706568189 Kent, Clark
1845139299 Mouse, Mickey
1279291359 Muffet, Miss
429164802 Sprat, Jack
*************************************


****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 2
****************************

*************************************
Enter patient first name: Donald
Enter patient last name: Duck
Enter date of birth (YYYY-MM-DD): 1954-01-22
*************************************
```

```
****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 1
****************************

*************************************
406943102 Duck, Donald
1706568189 Kent, Clark
1845139299 Mouse, Mickey
1279291359 Muffet, Miss
429164802 Sprat, Jack
*************************************


****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 3
****************************

*************************************
Enter patient first name: Miss
Enter patient last name: Muffet
Enter date of birth (YYYY-MM-DD): 2000-04-05
*************************************


****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 1
****************************
```

```
*************************************
406943102 Duck, Donald
1706568189 Kent, Clark
1845139299 Mouse, Mickey
429164802 Sprat, Jack
*************************************
```

```
****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 4
****************************
```

```
*************************************
Enter patient ID: 1706568189

Patient Information
ID: 1706568189
Last name: Kent
First name: Clark
DOB: 1953-06-12

Visits
2010-09-04
        Allergic to kryptonite :: Advised to stay away from keyptonite

*************************************
```

```
****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 5
****************************
```

```
*************************************
Enter patient ID: 406943102
What is the date of the visit (YYYY-MM-DD): 2018-08-2018
What was the reason for the visit?
Does not like the taste of quackers
What treatment was provided for the visit?
Gave him crackers instead

****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 4
****************************

*************************************
Enter patient ID: 406943102

Patient Information
ID: 406943102
Last name: Duck
First name: Donald
DOB: 1954-01-22

Visits
2018-08-2018
        Does not like the taste of quackers :: Gave him crackers instead

*************************************


****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: q
****************************
Changes were made to the database.
Do you want to save? (y/n):
y
****************************
BUILD SUCCESSFUL (total time: 3 minutes 50 seconds)
```

```
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: 1
****************************

*************************************
406943102 Duck, Donald
1706568189 Kent, Clark
1845139299 Mouse, Mickey
429164802 Sprat, Jack
*************************************


****************************
1. List all patient names
2. Add a new patient
3. Remove patient
4. Display patient information
5. Add a new visit for a patient
s. Save database
q. Quit Program
Choice from above: q
****************************
****************************
BUILD SUCCESSFUL (total time: 12 seconds)
```