Bulandi, Mark Lorence C204
Taytinf, Justine Jay 7OOP

Midterm Paired Task 1.

Object Oriented Analysis and Design

1. **Following the OO workflow as discussed in class**, you are task to design the OO Model of the given problem (use draw.io) of the scenario below:

Problem Statement. **Tiny Hospital** keeps information on **patients** and **hospital** rooms. The system assigns each patient a patient ID number. In addition, the patient's name and date of birth are recorded. Some patients are resident patients (they spend at least one night in the hospital) and others are outpatients (they are treated and released). Resident patients are assigned to a room. Each room is identified by a room number. The Tiny hospital system also stores the room type (private or semi-private) and room fee. Overtime, each room will have many patients who stay in it. Each resident patient will stay in only one room. The hospital system has features that can view patient information and view whether a room is occupied or not. Both patient and room entities must have features that allows adding, updating and searching of records.

STEP1. IDENTIFY all the necessary **OBJECT** within the problem domain

- Patient
- ResPatient
- OutPatient
- HosRoom
- HosFlow

STEP 2. IDENTIFY all the properties and methods/behaviors in the problem statement.

Patient

- patientID
- name
- dateOfBirth
- patientType

  Behavior
  - addPatient()
  - updatePatient()
  - searchPatientinfo()
  - getPatientinfo()

ResPatient

- roompatient
  - assignRoom()

OutPatient

  - discharge()

Bulandi, Mark Lorence C204
Taytinf, Justine Jay 7OOP

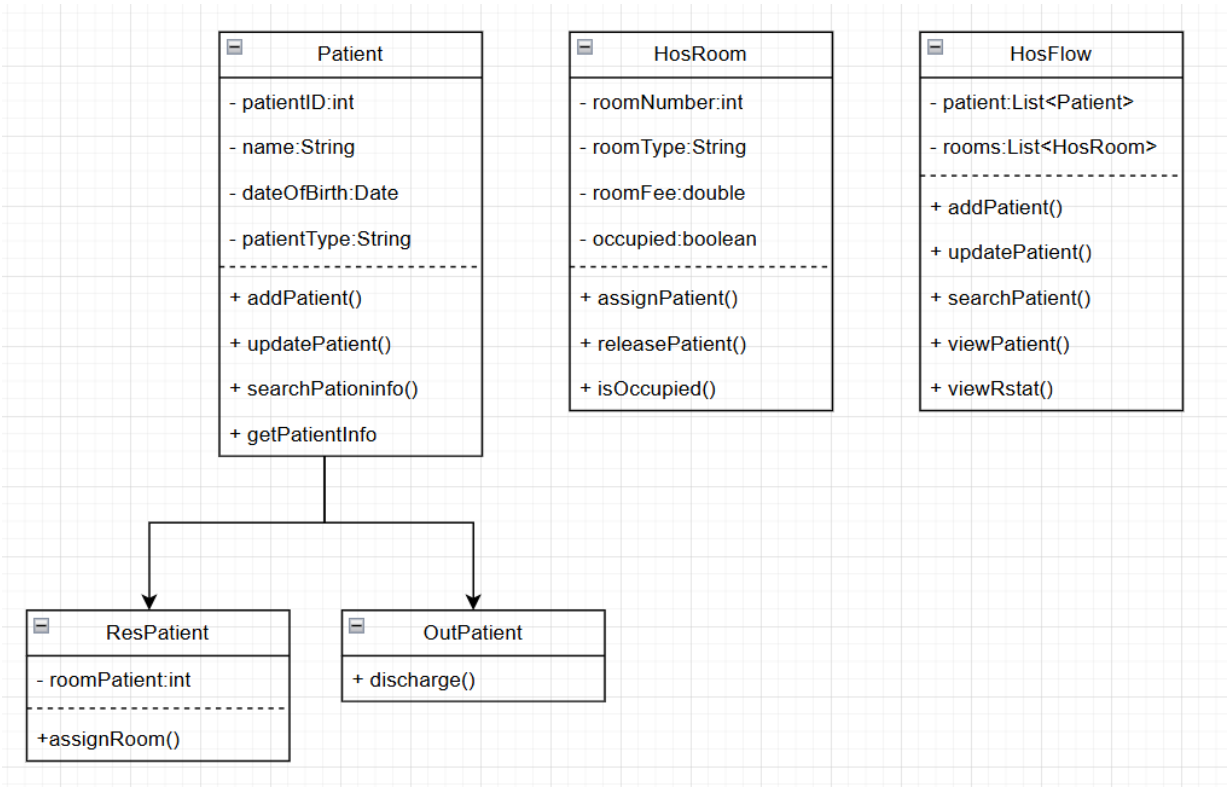
HosRoom

- roomNumber
- roomType
- roomFee
- occupied
    - assignPatient()
    - releasePatient()
    - isOccupied

HosFlow

- patientList
- roomList
    - addPatient()
    - updatePatient()
    - searchPatient()
    - viewPatient()
    - viewRstat()

STEP 3. Design the MODEL using a Class Diagram (You may use draw.io to represent the Blueprint of all the class that you need to create)



Bulandi, Mark Lorence C204
Taytinf, Justine Jay 7OOP

STEP 4. Implement the **class using Java code** construct of each interacting entities that you have identified.

```java
class Patient {
    int patientID;
    String name;
    Date dateOfBirth;
    String patientType;

    public Patient(int patientID, String name, Date dateOfBirth, String patientType) {
        this.patientID = patientID;
        this.name = name;
        this.dateOfBirth = dateOfBirth;
        this.patientType = patientType;
    }

    public void updatePatient(String name, Date dob) {
        this.name = name;
        this.dateOfBirth = dob;
    }

    public String getPatientInfo() {
        return "ID: " + patientID + ", Name: " + name + ", DOB: " + dateOfBirth + ", Type: " + patientType;
    }
}
```

```java
class HosRoom {
    int roomNumber;
    String roomType;
    double roomFee;
    boolean occupied = false;

    public HosRoom(int roomNumber, String roomType, double roomFee) {
        this.roomNumber = roomNumber;
        this.roomType = roomType;
        this.roomFee = roomFee;
    }

    public void assignPatient(Patient p) {
        occupied = true;
        System.out.println("Room " + roomNumber + " assigned to patient " + p.name);
    }

    public void releasePatient() {
        occupied = false;
        System.out.println("Room " + roomNumber + " is now available.");
    }

    public boolean isOccupied() {
        return occupied;
    }
}
```

```java
class ResPatient extends Patient {
    int roomPatient;

    public ResPatient(int patientID, String name, Date dob, int roomPatient) {
        super(patientID, name, dob, "Resident");
        this.roomPatient = roomPatient;
    }

    public void assignRoom(HosRoom room) {
        this.roomPatient = room.roomNumber;
        room.assignPatient(this);
    }
}
```

```java
class HosFlow {
    List<Patient> patients = new ArrayList<>();
    List<HosRoom> rooms = new ArrayList<>();

    public void addPatient(Patient p) {
        patients.add(p);
    }

    public void updatePatient(int id, String newName, Date newDob) {
        for (Patient p : patients) {
            if (p.patientID == id) {
                p.updatePatient(newName, newDob);
                System.out.println("Updated patient " + id);
                return;
            }
        }
        System.out.println("Patient not found.");
    }

    public Patient searchPatient(int id) {
        for (Patient p : patients) {
            if (p.patientID == id) {
                return p;
            }
        }
        return null;
    }

    public void viewPatient(int id) {
        Patient p = searchPatient(id);
        if (p != null) {
            System.out.println(p.getPatientInfo());
        } else {
            System.out.println("Patient not found.");
        }
    }

    public void viewRstat(int roomNumber) {
        for (HosRoom r : rooms) {
            if (r.roomNumber == roomNumber) {
                System.out.println("Room " + r.roomNumber + " occupied: " + r.isOccupied());
                return;
            }
        }
        System.out.println("Room not found.");
    }
}
```

```java
class OutPatient extends Patient {
    public OutPatient(int patientID, String name, Date dob) {
        super(patientID, name, dob, "Outpatient");
    }

    public void discharge() {
        System.out.println(name + " has been discharged.");
    }
}
```

```java
public class Main {
    public static void main(String[] args) {
        HosFlow system = new HosFlow();

        HosRoom room101 = new HosRoom(101, "Private", 1500.0);
        HosRoom room102 = new HosRoom(102, "Semi-Private", 1000.0);
        system.rooms.add(room101);
        system.rooms.add(room102);

        ResPatient rp = new ResPatient(1, "Juan Dela Cruz", new Date(), 101);
        OutPatient op = new OutPatient(2, "Maria Clara", new Date());

        system.addPatient(rp);
        system.addPatient(op);

        rp.assignRoom(room101);

        system.viewPatient(1);
        system.viewPatient(2);

        system.viewRstat(101);
        system.viewRstat(102);

        op.discharge();

        room101.releasePatient();
        system.viewRstat(101);
    }
}
```

Note: Highlight all the outputs following the example from STEP 1 to STEP 4 as shown in the lecture