LAB 211 Assignment

Type:	**Long Assignment**
Code:	**J1.L.P0018**
LOC:	**500**
Slot(s):	**N/A**

Title

Hospital Management

Background

A hospital needs a program to **manage nurse and patient information**. With basic requirements such as **creating** a nurse (or a patient), **display** nurse (or patient) information and **updating** their information. Nurse and patient 's information is **stored in a text or binary file** (**nurse**.dat). **Using HashMap structure to manage nurses and patients**

Program Specifications

Build a **hospital management program**. With the following basic functions:

1. **Nurse's management**
   1.1 **Create** a nurse
   1.2 **Find** the nurse
   1.3 **Update** the nurse
   1.4 **Delete** the nurse
2. **Patient's management**
   2.1 **Add** a patient
   2.2 **Display** patients
   2.3 **Sort** the patients list
   2.4 **Save** data
   2.5 **Load** data

   Others – Quit.

Each menu choice should invoke an appropriate function to perform the selected menu item. Your program must **display the menu after each task** and **wait for the user** to select another option until the user chooses to quit the program.

Define a **Person class** with properties such as **id, name, age, gender, address, phone, etc**.

Create a **Patient class** that **extends** the **Person** class, with additional properties such as **diagnosis, admissionDate, dischargeDate, nurseAssigned, etc.**

Create a **Nurse class** that also **extends** the **Person** class, with additional properties such as **staffID, department, shift, salary, etc.**

Create a **Hospital class** that **manages** the **Patient** and **Nurse** records.

***Note: each patient is cared by two nurse and one nurse takes care of maximum 2 patients.**

Features:

Function 1. Create a nurse – 50 LOC

- User is **required** inputting a nurse: **staffID, name, age, gender, address, phone, department, shift, salary**.
- The system should **check the valid data** with the following conditions:
  - **All** fields are **not allowed null**.
  - The **staffID** field must be **unique**.
  - The **length of the department** field must be from **3 to 50 characters**.
  - The import **phone** field must be a **valid phone format**.
  - The **age** field must be a **positive number**.
  - The **salary** field must be a **positive number**.
- **Add** the nurse **to** the **collection of nurses**.
- **Ask** to continue adding a new nurse or go back to the main menu.

Function 2. Find a nurse – 50 LOC

- User is **required** inputting the **nurse 's name** or **part of the name**.
- If the nurse **does not exist**, the message "**The nurse does not exist**" is displayed. **Otherwise**, display the nurse.

Function 3. Update a nurse – 50 LOC

- User is **required** inputting the **staffID**.
- If the nurse **does not exist**, the message "**The nurse does not exist**" is displayed. **Otherwise**, the user can edit the nurse.
- The application should **show the result** of the update: **success or failure**.

Function 4. Delete a nurse – 50 LOC

- User is **required** inputting the **staffID**.
- If the nurse **does not exist**, the message "**The nurse does not exist**" is displayed. **Otherwise**, the user can delete the nurse.
- The application must **show the confirmation message** before deleting.
- The nurse **cannot be deleted** if she **has a task** (look after **a patient specification in below**).
- The application should **show the result** of the deletion: **success or failure**

Function 5. Add a patient – 50 LOC

- The program **requires** user to **input** a **piece of patient information** including **id, name, age, gender, address, phone, diagnosis, admissionDate, dischargeDate, nurseAssigned.**
- The system should **check the valid data** with the following conditions:

- **All** fields are **not allowed null**.
- The **id** fields must be **unique**.
- The **admissionDate, dischargeDate** fields must be a **valid date format**.
- The **age** field must be a **positive integer**.
- The import **phone** field must be **a valid phone format**.
- The **nurseAssigned** field must be **on the list of available nurses**.

– **Add** the **patient to** the **collection** of patients.

– **Ask** to continue adding a new patient or go back to the main menu.

Function 6. Display patients – 50 LOC

– User is **required** inputting a **start and end date**.

– The application should **show patients information** based on **typed date range** below if applicable (admission date).

LIST OF PATIENTS
Start date: 01/01/2023
End date: 21/04/2023

No.	Patient Id	Admission Date	Full name	Phone	Diagnosis
1	P0006	10/01/2023	Cir Smith	0939064869	Flu
2	P0007	10/04/2023	Bill Jamie	0932123495	Fever
3	P0008	11/04/2023	Ann Smith	0853321452	Cough
4	P0009	20/04/2023	Ronaldo Delima	0273456910	flu

Function 7. Sort patients – 50 LOC

– User is **required** inputting a **sorted field** (patient **id** or patient's **name**) and the **sort patient** (ASC, DESC).

– The system will **sort and display** the collection of patients as below.

LIST OF PATIENTS
Sorted by: patient's name
Sort order: ASC

No.	Patient Id	Admission Date	Full name	Phone	Diagnosis
3	P0008	11/04/2023	Ann Smith	0853321452	Cough
2	P0007	10/04/2023	Bill Jamie	0932123495	Fever
1	P0006	10/01/2023	Cir Smith	0939064869	Flu
4	P0009	20/04/2023	Ronaldo Delima	0273456910	flu

Function 8. Save data – 50 LOC

– The application should **save** the **collection of nurses** to the **binary file** that is named as **nurses.dat**.

– The application should **save** the **collection of patients** to the **binary file** that is named as **patients.dat.**

Function 9. Load data – 50 LOC

- The application will **load** the **collection of nurses** from **the nurses.dat file**.
- The application will **load** the **collection** of **patients** from the **patients.dat file**.

Function 10. Quit – 50 LOC

- **Exit** the program.
- The application **must show** the **confirmation** message before exiting.
- The application **must save** data to files if data **has changed**.

The above specifications are only **basic information**; you must perform a requirements **analysis step and build the application according to real requirements.**

The lecturer will explain the requirement only once in the first slot of the assignment.