**Overview**

This system is designed to manage a hospital network consisting of five independent branches, supporting patient management, medical staff allocation, medication procurement, financial settlement, and other functionalities. The following outlines the class design and its detailed description.

**1. Pharmacy Class**

**Data Members**

**Private Data:**

* int pharmacyID: Unique identifier for the pharmacy
* double totalBill: Total billing amount of the pharmacy

**Member Functions**

**Public Functions:**

* Pharmacy(int id): Constructor to initialize the pharmacy ID
* void addBill(double amount): Records medication purchase bills
* double getTotalBill() const: Retrieves the total billing amount of the pharmacy

**2. Medical Staff Base Class**

**Data Members**

**Protected Data:**

* int staffID: Unique identifier for the medical staff
* std::string name: Name of the medical staff
* int assignedHospital: ID of the assigned hospital branch

**3. Doctor Class**

**Data Members**

**Private Data:**

* std::vector<int> assignedPatients: List of assigned patient IDs

**Member Functions**

**Public Functions:**

* bool assignPatient(int patientID): Assigns a patient (checks the maximum consultation capacity)
* void releasePatient(int patientID): Releases a patient from the doctor's care

**4. Nurse Class**

**Data Members**

**Private Data:**

* std::vector<int> assignedPatients: List of assigned patient IDs (maximum of 2)

**Member Functions**

**Public Functions:**

* bool assignPatient(int patientID): Assigns a patient (strict limit of 2 patients per nurse)
* void releasePatient(int patientID): Releases a patient from the nurse's care

**5. Patient Class**

**Data Members**

**Private Data:**

* int patientID: Unique identifier for the patient
* int currentHospitalID: ID of the current hospital branch
* std::string personalInfo: Personal information of the patient
* std::string medicalCondition: Medical condition of the patient
* std::vector<std::string> treatments: List of treatment plans
* int attendingDoctorID: ID of the primary attending doctor
* std::vector<int> consultingDoctors: List of consulting doctor IDs

**Member Functions**

**Public Functions:**

* Patient(int id, std::string info): Constructor to initialize patient ID and personal information
* void updateCondition(std::string condition): Updates medical condition records
* void addTreatment(std::string treatment): Adds a treatment plan
* void transferHospital(int newHospitalID): Transfers the patient to another hospital branch

**6. Hospital Branch Class**

**Data Members**

**Private Data:**

* int branchID: Unique identifier for the hospital branch
* std::vector<Patient> patients: List of current patients in the branch (maximum of 20)
* std::vector<Doctor> doctors: List of doctors in the branch (at least 3)
* std::vector<Nurse> nurses: List of nurses in the branch (at least 5)
* std::unordered\_map<int, double> pharmacyBills: Records of pharmacy bills

**Member Functions**

**Public Functions:**

* bool admitPatient(Patient& p, int doctorID): Admits a patient (allocates bed and assigns primary doctor)
* bool dischargePatient(int patientID): Processes patient discharge (verifies discharge authorization)
* bool assignDoctor(int patientID): Assigns a primary doctor to the patient
* bool assignNurse(int patientID): Assigns a nurse to the patient
* void purchaseMedication(int pharmacyID, double amount): Processes medication procurement payment
* int getAvailableBeds() const: Retrieves the number of available beds
* void displayDailyReport() const: Generates a daily report

**7. Hospital Management System Core Class**

**Data Members**

**Private Data:**

* std::vector<HospitalBranch> branches: List of 5 hospital branches
* std::vector<Pharmacy> pharmacies: List of 20 pharmacies
* std::unordered\_map<int, Patient> allPatients: Records of all patients
* static int nextPatientID: ID generator for patients

**Member Functions**

**Public Functions:**

* void initializeSystem(): Initializes system resources (5 branches, 20 pharmacies, medical staff)
* int registerPatient(std::string info): Registers a new patient (generates a unique ID)
* bool transferPatient(int patientID, int newBranch): Transfers a patient across hospital branches
* void addDoctor(int branchID, Doctor doc): Adds a doctor to the specified hospital branch
* void addNurse(int branchID, Nurse nrs): Adds a nurse to the specified hospital branch
* void addPharmacy(Pharmacy pharma): Registers a pharmacy
* void generateFinancialReport() const: Generates a financial report