**Library Management System Design a C++ program to manage a library system that includes books, members, and membership types. The system should include:**

1. **Book Class**:
   * Each book has an ID, title, author, and status (available, issued).
   * Implement a static member to count the total number of books.
   * Include methods to get book details and update its status.
   * Implement a friend function to display the details of a given book, including the static count of all books.
2. **Member Class**:
   * Each member has a member ID, name, and a list of borrowed books.
   * Implement methods to borrow and return books, ensuring the book's status is updated correctly.
   * Include a method to display member details and the list of borrowed books.
3. **RegularMember Class** (Intermediate Derived Class):
   * Inherits from the Member class.
   * Adds a limit on the number of books a member can borrow at a time.
   * Implement methods to enforce the borrowing limit.
4. **PremiumMember Class** (Final Derived Class):
   * Inherits from the RegularMember class.
   * Adds additional features for premium members, such as an extended borrowing limit and access to exclusive books.
   * Implement methods to handle the special privileges of premium members.

**Requirements:**

* Implement **Book**, **Member**, **RegularMember**, and **PremiumMember** classes with appropriate attributes and methods.
* Demonstrate **multilevel inheritance** by having **PremiumMember** inherit from **RegularMember**, which in turn inherits from **Member**.
* Track the number of available books using a static data member in the **Book** class.
* Implement a **friend function** in the **Book** class to access private members and display book details.
* Ensure **dynamic memory management** for **Book** and **PremiumMember** objects.
* Include **constructors and destructors** for proper resource management and initialization.

**Hospital Management System Design a C++ program to manage a hospital system that includes doctors, patients, and different types of patient care. The system should include:**

1. **Doctor Class**:
   * Each doctor has a unique ID, name, specialization, and status (e.g., available, busy).
   * Implement a static member to count the total number of doctors.
   * Include methods to get doctor details and update status.
   * Implement a friend function to display the details of a given doctor, including the static count of all doctors.
2. **Patient Class**:
   * Each patient has a unique ID, name, illness, and status (e.g., under treatment, discharged).
   * Implement methods to assign doctors, receive treatment, and discharge the patient.
   * Include a method to display patient details and the current doctor assigned.
3. **Outpatient Class** (Intermediate Derived Class):
   * Inherits from the **Patient** class.
   * Adds additional attributes for appointment times and consultation records.
   * Implement methods to schedule appointments and manage outpatient consultations.
4. **Inpatient Class** (Final Derived Class):
   * Inherits from the **Outpatient** class.
   * Adds additional features for inpatients, such as room allocation and extended care.
   * Implement methods to handle room assignments, extended care, and hospital discharge procedures.

**Requirements:**

* Implement **Doctor**, **Patient**, **Outpatient**, and **Inpatient** classes with appropriate attributes and methods.
* Demonstrate **multilevel inheritance** by having **Inpatient** inherit from **Outpatient**, which in turn inherits from **Patient**.
* Track the number of available doctors using a static data member in the **Doctor** class.
* Implement a **friend function** in the **Doctor** class to access private members and display doctor details.
* Ensure **dynamic memory management** for **Doctor** and **Inpatient** objects.
* Include **constructors and destructors** for proper resource management and initialization.