#### **Test 1**

#### **Problem Statement**

You are tasked with designing a **Dynamic Supply Chain Management System** for a company. The system must manage suppliers, products, and orders, ensuring that complex constraints and dynamic behavior are handled effectively. The solution must leverage abstraction, interfaces, and polymorphism to ensure scalability, maintainability, and modularity.

### **Detailed Requirements**

1. **Supplier and Product Relationships** (30 marks):
   * Suppliers:
     + Each supplier has a unique ID, name, quality rating (1–5), and lead time (in days).
     + Suppliers provide a list of products they offer, each with its own price.
     + Suppliers may impose specific constraints, such as minimum order quantities or product-specific lead times.
   * Products:
     + Products have a unique ID, name, category, and unit price.
     + A product can be supplied by multiple suppliers, and the system must identify the "best supplier" dynamically based on:
       - Lowest cost (primary factor).
       - Highest quality rating (secondary factor if costs are tied).
       - Shortest lead time (final factor for ties).
     + The selection logic for the best supplier must be extensible (e.g., to include discounts or special offers in the future).
2. **Order Processing** (30 marks):
   * Orders:
     + Orders consist of multiple products and their quantities.
     + When processing an order, the system must:
       - Verify product availability with suppliers.
       - Select the best supplier for each product based on the logic in (1).
       - Generate an order summary showing total cost, estimated delivery time, and supplier details.
     + Partial Fulfillment:
       - If a product is unavailable with the selected supplier, attempt to procure it from alternative suppliers.
       - If no supplier can fulfill the product, mark it as "backordered" and notify the user.
   * Tracking:
     + Orders must maintain a status: Pending, Fulfilled, Partially Fulfilled, or Backordered.
     + Provide detailed logs of the allocation process, showing which supplier fulfilled each product and the cost associated with each fulfillment.
3. **Dynamic Constraints** (20 marks):
   * Product Dependencies:
     + Certain products cannot be ordered without others (e.g., Product A requires Product B).
     + Implement logic to validate these dependencies dynamically during the ordering process.
   * Supplier Constraints:
     + Some suppliers have minimum order quantities or deliver only to specific regions.
     + The system must validate these constraints when processing orders and provide clear error messages if constraints are violated.
4. **Extensibility and Scalability** (20 marks):
   * Future Changes:
     + Allow adding new supplier selection criteria, such as promotional discounts or priority partnerships, with minimal impact on existing code.
     + Support new product categories or supplier constraints dynamically without rewriting existing functionality.
   * Reporting:
     + Implement a reporting mechanism that provides summaries like:
       - Most frequently ordered products.
       - Most reliable suppliers (based on fulfillment success rate).
       - Total revenue generated for each supplier.

**Test 2**

#### **Problem Statement**

Design a **Hospital Management System** that organizes and tracks doctors, patients, and medical appointments. The system must handle different types of appointments, enforce doctor availability, and validate patient medical histories. The solution should emphasize abstraction, interfaces, and adherence to the SOLID principles for a modular and scalable design.

### **Detailed Requirements**

1. **Doctor and Patient Management** (30 marks):
   * **Doctors**:
     + Each doctor has a specialization (e.g., cardiologist, dermatologist) and a schedule of availability (time slots).
     + Doctors may have restrictions, such as the maximum number of patients they can see per day.
   * **Patients**:
     + Each patient has a unique ID, medical history (e.g., chronic conditions, allergies), and a list of past and upcoming appointments.
     + The system must validate that the selected doctor is suitable for the patient based on their specialization and the patient's medical history.
2. **Appointment Scheduling** (30 marks):
   * Patients can book appointments with doctors based on their availability.
   * The system must enforce the following constraints:
     + A doctor cannot have overlapping appointments.
     + Appointments must respect the maximum number of patients a doctor can see in a day.
     + Certain types of patients (e.g., emergency cases) may bypass normal constraints and take priority in the schedule.
   * Provide a mechanism to reschedule or cancel appointments dynamically, ensuring no conflicts.
3. **Dynamic Constraints** (20 marks):
   * Appointment types include:
     + **Routine Check-Up**: Regular appointment with no special priority.
     + **Emergency**: Requires immediate scheduling, even if it overrides normal constraints.
     + **Follow-Up**: Must occur within a specific time window after a previous appointment.
   * The system must validate appointment rules dynamically based on type and provide clear feedback if constraints are violated.
4. **Extensibility and Scalability** (20 marks):
   * **Future Changes**:
     + Allow adding new appointment types or constraints with minimal changes to the core logic.
     + Support new features such as teleconsultations or specialist referrals dynamically.
   * **Reporting**:
     + Implement reporting to generate summaries, such as:
       - Most frequently booked doctors.
       - Average number of appointments per day.
       - Patients with the most emergency cases.