Project: University Cafeteria Order & Loyalty System

Team size 3 to 5 from the same group allocation, under the same mentor. No cross-group teams. If you want to have bigger to smaller team size discuss with your mentor.

This project aims to develop a straightforward system for managing orders at a university cafeteria and implementing a basic loyalty program for students. It's a familiar concept that students can easily understand, reducing the cognitive load on grasping the problem domain and allowing them to focus more on the implementation details.

Core Functional Requirements:

1. User (Student) Management:

- **FR1.1:** The system shall allow students to register with a unique ID and create a simple profile (name, student ID).
- **FR1.2:** The system shall allow students to log in securely.
- **FR1.3:** The system shall display the student's current loyalty points balance.

2. Menu Management:

- **FR2.1:** The system shall allow cafeteria staff (admin user) to add, edit, and remove menu items.
- **FR2.2:** Each menu item shall have a name, description, price, and category (e.g., "Main Course," "Snack," "Drink").
- **FR2.3:** The system shall display the available menu to students.

3. Order Placement:

- **FR3.1:** Students shall be able to browse the menu and select items to add to their order.
- **FR3.2:** The system shall display the total cost of the current order.
- **FR3.3:** Students shall be able to confirm and place an order.
- **FR3.4:** The system shall generate a unique order ID for each placed order.

4. Loyalty Program:

- **FR4.1:** The system shall award loyalty points to students based on their order value (e.g., 1 point for every EGP 10 spent).
- **FR4.2:** The system shall allow students to redeem loyalty points for discounts or free items (e.g., 100 points for a free coffee, or EGP 10 discount for 50 points).
- **FR4.3:** The system shall deduct points from the student's balance upon redemption.

5. Order Fulfillment (Cafeteria Staff View):

- **FR5.1:** The system shall display a list of pending orders for cafeteria staff.
- **FR5.2:** Staff shall be able to mark orders as "Preparing" and "Ready for Pickup."

- **FR5.3:** The system shall notify the student when their order is "Ready for Pickup" (e.g., in-app notification).

6. Basic Reporting:

- **FR6.1:** The system shall allow cafeteria staff to view a summary of daily/weekly sales and loyalty point redemptions.

How this project incorporates the requested elements (Simplified Scope):

- **Advanced OOP Design (Accessible Level):**
    - **Classes:** Student, MenuItem, Order, LoyaltyProgram, MenuManager, OrderProcessor. These are clearly defined and relate directly to real-world entities.
    - **Encapsulation:** Each class will encapsulate its data (e.g., MenuItem encapsulates name, price, category) and behavior (e.g., Order calculates total, LoyaltyProgram adds/redeems points).
    - **Association:** Student will have an association with Orders, Orders will have an association with MenuItems.
- **Strict Adherence to SOLID Principles (Focus on the most applicable ones for this scope):**
    - **Single Responsibility Principle (SRP):** This is highly applicable here.
        - StudentManager class handles student registration/login.
        - MenuManager class handles adding/editing/removing menu items.
        - OrderProcessor class handles placing and tracking orders.
        - LoyaltyProgram class handles point calculation and redemption.
          Each class does one thing well.
    - **Open/Closed Principle (OCP):**
        - If a new type of loyalty reward is introduced (e.g., a tiered system), the LoyaltyProgram could be extended or use a strategy pattern for point calculation without modifying the OrderProcessor.
        - If a new payment method is introduced, it could be added without altering the core OrderProcessor logic (e.g., via an IPaymentProcessor interface).
    - **Dependency Inversion Principle (DIP):**
        - The OrderProcessor should depend on an IMenuProvider interface rather than a concrete MenuManager class. This allows swapping out how menus are sourced (e.g., from a database vs. hardcoded list) without affecting the OrderProcessor.
        - The LoyaltyProgram might depend on an IStudentRepository interface.
- **Abbott’s Technique (Simplified Application):**

    - **Nouns:** Student, Cafeteria, Menu, MenuItem, Order, Point, Discount, ID, Price, Category, Staff, Report. These clearly map to potential classes or attributes.
    - **Verbs:** Register, Login, Add, Edit, Remove, Display, Browse, Select, Place, Confirm, Award, Redeem, Deduct, Mark, Notify, View, Summarize. These become methods for the identified classes.
    - **CRC Cards (Simplified):** Students can easily create CRC cards for Student, MenuItem, Order, and LoyaltyProgram. For example:
        - **Class:** Order
        - **Responsibilities:** Holds selected MenuItems, Calculates total price, Generates order ID, Tracks status.
        - **Collaborators:** MenuItem, LoyaltyProgram (to award points), OrderProcessor (to be managed by).
- **Current Tech Trends and Real-World Application (Simplified Scope):**
    - **Digital Ordering:** Common in modern food service.
    - **Loyalty Programs:** Ubiquitous in retail.
    - **Basic Data Management:** Storing and retrieving information.
    - **User Interface (Even basic console or simple GUI):** Provides a tangible interaction point.
    - **Notification Systems:** A simple in-app notification for order pickup.

- Adopt all concepts discussed in the java core course, including streams in the design and coding for your application.
- Develop as a consol app, but if you used JFX graphics, you will get bonus.

📊 Grading Criteria

Evaluation Item	Weight
OOP Design & SOLID Principles	30%
Abbott’s Analysis & Scenario Modeling	20%
Design Patterns & Architecture	20 %
Code Clarity, Testing & Documentation	10%
Advanced Java core Concepts	20%
