1. **Library Management System:**

- Create classes for `Book` (title, author, ISBN) and `Member` (name, ID).
- Design a `Library` class with methods to:
    - Add books (handle duplicates). (Use Set)
    - Issue books to members (track borrowed books using a List).
    - Return books.
    - Search for books by title or author (return matching books as a List).
    - Display all available books.
- Implement user interaction with options to add/remove books, manage members, borrow/return books, and search the library.

2. **Social Network Simulator:**

- Develop classes for `User` (name, profile pic) and `Post` (content, timestamp).
- Create a `SocialNetwork` class to manage users and their posts.
    - Implement methods for:
        - Adding users.
        - Creating and posting content.
        - Following other users (use a Set to store followed users).
        - Displaying a user's feed (show posts from themself and followed users).
- Implement a user interface for users to create profiles, post content, follow others, and view their feed.

3. **Inventory Management System:**

- Define classes for `Product` (name, price, quantity) and `Order` (items, customer details).
- Design a `Store` class with functionalities:
    - Add products to inventory (use a List to maintain order).
    - Update product quantity upon purchase. (Use a Set to track unique products)
    - Create orders (add items to order, update inventory).
    - Display current inventory and order history.
- Implement user interaction for adding products, creating orders, and viewing inventory/orders.

4. **Course Registration System:**

- Create classes for `Course` (name, instructor) and `Student` (name, ID).
- Design a `RegistrationSystem` class with methods to:
    - Manage courses offered. (Use a List)
    - Enroll students in courses (handle capacity limits, use a Set for enrolled students per course).
    - Withdraw students from courses.
    - Display course information and enrolled students.

- Implement user interaction for adding courses, enrolling/withdrawing students, and viewing course details.

5. **To-Do List Application:**

- Define a class `Task` (description, due date, priority).
- Create a `ToDoList` class with functionalities:
    - Add tasks (categorized by priority, use a separate List for each priority level).
    - Mark tasks as completed/incomplete.
    - Edit task details.
    - Display all tasks categorized by priority and completion status.
- Implement user interaction for adding/editing tasks, marking completion, and viewing the list.