**Task 1: Inventory Management System with ArrayLists**

**Description**: Create an inventory management system for a store using ArrayList. The system should allow the user to:

1. Add new items to the inventory. Each item should have a name, quantity, and price.
2. Update the quantity of an existing item.
3. Remove an item from the inventory.
4. Search for an item by name and display its details.
5. Calculate and display the total value of the inventory.

**Desired Output**:

1. Add Item

2. Update Item Quantity

3. Remove Item

4. Search Item

5. Display Inventory Value

6. Exit

Enter your choice: 1

Enter item name: Widget

Enter item quantity: 10

Enter item price: 2.50

Item added successfully.

1. Add Item

2. Update Item Quantity

3. Remove Item

4. Search Item

5. Display Inventory Value

6. Exit

Enter your choice: 5

Total inventory value: $25.00

**Steps**:

1. Define an Item class with attributes for name, quantity, and price.
2. Create an ArrayList<Item> to store the inventory items.
3. Implement methods to add, update, remove, and search items.
4. Implement a method to calculate the total value of the inventory.
5. Create a menu-driven interface to interact with the inventory system.

**Task 2: Student Grades System with LinkedLists**

**Description**: Create a student grades system using LinkedList. The system should allow the user to:

1. Add a student and their grades (multiple subjects).
2. Update a student's grade for a specific subject.
3. Remove a student from the system.
4. Display a student's average grade.
5. Display all students sorted by their average grade in descending order.

**Desired Output**:

1. Add Student

2. Update Grade

3. Remove Student

4. Display Student Average

5. Display All Students Sorted by Average

6. Exit

Enter your choice: 1

Enter student name: John Doe

Enter number of subjects: 3

Enter grade for subject 1: 85

Enter grade for subject 2: 90

Enter grade for subject 3: 78

Student added successfully.

1. Add Student

2. Update Grade

3. Remove Student

4. Display Student Average

5. Display All Students Sorted by Average

6. Exit

Enter your choice: 4

Enter student name: John Doe

John Doe's average grade: 84.33

**Steps**:

1. Define a Student class with attributes for name and a LinkedList<Integer> for grades.
2. Create a LinkedList<Student> to store the students.
3. Implement methods to add students, update grades, remove students, and calculate average grades.
4. Implement a method to sort and display students by their average grade.
5. Create a menu-driven interface to interact with the student grades system.

**Task 1: Hospital Management System with Arrays and ArrayLists**

**Description**: Create a hospital management system that manages doctors and patients. The system should allow the user to:

1. Add a doctor with their specialty.
2. Add a patient with their assigned doctor.
3. Display all patients assigned to a specific doctor.
4. Display all doctors and their assigned patients.
5. Remove a patient from the system.
6. Transfer a patient to a different doctor.

**Desired Output**:

1. Add Doctor

2. Add Patient

3. Display Patients of a Doctor

4. Display All Doctors and Patients

5. Remove Patient

6. Transfer Patient

7. Exit

Enter your choice: 1

Enter doctor name: Dr. Smith

Enter specialty: Cardiology

Doctor added successfully.

1. Add Doctor

2. Add Patient

3. Display Patients of a Doctor

4. Display All Doctors and Patients

5. Remove Patient

6. Transfer Patient

7. Exit

Enter your choice: 2

Enter patient name: John Doe

Enter assigned doctor: Dr. Smith

Patient added successfully.

1. Add Doctor

2. Add Patient

3. Display Patients of a Doctor

4. Display All Doctors and Patients

5. Remove Patient

6. Transfer Patient

7. Exit

Enter your choice: 3

Enter doctor name: Dr. Smith

Patients of Dr. Smith:

- John Doe

**Steps**:

1. Define Doctor and Patient classes.
2. Create an ArrayList<Doctor> to store doctors and an ArrayList<Patient> to store patients.
3. Implement methods to add doctors and patients.
4. Implement methods to display patients of a specific doctor and all doctors with their patients.
5. Implement methods to remove and transfer patients.
6. Create a menu-driven interface to interact with the hospital management system.