Mock Practical Questions for Intermediate Python Certification Exam

[10 Marks each]

1. Design an application to track daily expenses using comprehension method. Your application should allow users to:
    a. Add expenses with details (amount, category, date).
    b. View all expenses in a formatted manner.
    c. Filter expenses by category using list comprehension.
    d. Generate a summary report showing total expenses by category.

2. Create a task management application that allows users to add, view, and delete tasks. Implement the following features using decorators:
    a. Logging Decorator: This decorator should log each action performed (adding, viewing, deleting tasks), including the action type, the task description, and the timestamp.
    b. Input Validation Decorator: This decorator should ensure that task descriptions are valid (i.e., non-empty strings) before allowing the task to be added or deleted.
    c. Requirements:
        i. The application should provide a simple text-based menu for user interaction.
        ii. Implement functions for adding tasks, viewing the current list of tasks, and deleting tasks.
        iii. Log each action to the console.
    d. Example Functionality:
        i. When a user adds a task, it should log the action.
        ii. If a user tries to add an empty task, it should raise a validation error.
        iii. When viewing tasks, it should display the current list of tasks.

3. Develop a library management system using comprehension method that allows users to manage books. Your system should include the following features:
    a. Add books with title, author, and ISBN.
    b. Search for books by title or author using dictionary comprehension.
    c. Delete books from the inventory.
    d. Display all books in a user-friendly format.

4. Explain how the `RecipeManager` class works, including its methods for adding recipes, searching for recipes by ingredient, and displaying all recipes with the help of code. Discuss the data structures used and any important features such as ingredient uniqueness. [Use frozen-set]

5. Create a small application that processes a list of sales data. Each entry in the sales data contains a product name, quantity sold, and price per item. Your task is to calculate the total revenue generated from products that meet the following criteria. [Use higher order functions and functional programming]
    a. Only include products where the quantity sold is greater than 10.
    b. Apply a discount of 10% on products priced above ₹50.
    c. Return the total revenue as well as a list of products with their respective revenues after applying the discount if applicable.

6. Create a simple logging decorator that logs the execution time of a function. Use this decorator to time a function that simulates processing data (for example, by sleeping for a specified number of seconds). The function should take an integer parameter representing the number of seconds to sleep. After applying the decorator, the output should display both the execution time and the result of the function. [Use decorators]