DSE 2144– Advanced Programming Language Lab
Week 6 – Date: 13th September 2024

Exercise 1:

You are tasked with designing a **Library Management System** for a local library. The library has both **EBooks** (digital format) and **Printed Books** (physical copies), and the system should allow members (both **students** and **teachers**) to borrow books. The library also has a **librarian** who manages the addition and removal of books.
Your system should include the following features:

1. **Book Management**:
    - Books can be either **EBooks** or **Printed Books**.
    - Each book should have a **title**, **author**, and **ISBN**.
    - EBooks should have a **file format**, while Printed Books should have a **page count**.
2. **Member Management**:
    - The library has **members** who can either be **students** or **teachers**.
    - Each member has a **name** and **member ID**.
    - Members should be able to **borrow books**.
3. **Librarian Management**:
    - A **librarian** can add or remove books from the library.
    - A librarian can be both a **student** and a **teacher**.
4. **Library Operations**:
    - The system should allow the librarian to:
        - **Add new books** to the library.
        - **Remove books** from the library using their ISBN.
        - **Search for books** by title or author.

Requirements:
Using Python and Object-Oriented Programming principles, implement the following:

1. Create a class hierarchy to represent **books** (including EBooks and Printed Books).
2. Create a class hierarchy to represent **members** (students and teachers).
3. Implement the functionalities for adding, removing, and searching for books.
4. Demonstrate the following types of inheritance:
    - **Single Inheritance** for books.
    - **Multiple Inheritance** for the librarian, who is both a student and a teacher.
    - **Hierarchical Inheritance** for members (students and teachers).

Tasks:

1. **Book Management**:
    - Define a base class Book with attributes for title, author, and ISBN.
    - Define a subclass EBook that adds the attribute for file format.
    - Define another subclass PrintedBook that adds the attribute for page count.
2. **Member and Librarian Management**:
    - Define a base class Member with attributes for name and member ID.
    - Define two subclasses: Student and Teacher, which inherit from Member.
    - Create a Librarian class that inherits from both Student and Teacher (multiple inheritance).

3. **Library Class**:

    - Implement a Library class to manage the collection of books.
    - Add methods to the Library class to:
        - Add new books.
        - Remove a book by its ISBN.
        - Search for books by title or author.
4. **Demonstration**:
    - Instantiate a library and add books (both EBooks and Printed Books) to it.
    - Demonstrate searching for books using keywords.
    - Show how a librarian can add and remove books from the system.

Exercise 2: Advanced E-Commerce System Utilizing Polymorphism

A rapidly growing online retail company is looking to upgrade its **E-Commerce System**. They need the system to manage various types of products, allow users to add them to their shopping carts, apply discounts, and handle different payment methods. To make the system robust, flexible, and scalable, you decide to use **Object-Oriented Programming (OOP)** principles.

Objectives:

1. **Product Management**:
    - Products in the system belong to different categories such as **Electronics** and **Clothing**. Each product category has its own discount logic.
    - Discounts should be applied based on product types, demonstrating **method overriding**.
2. **Shopping Cart Management**:
    - Users should be able to add multiple items to their shopping carts and see the total cost after discounts.
    - The system should allow **merging two shopping carts** using operator overloading.
3. **Payment Processing**:
    - Customers should be able to process payments using various methods (e.g., credit card, PayPal). Even though Python does not natively support **method overloading**, it should be simulated to handle different payment methods efficiently.

Functional Requirements:

1. **Products**:
    - Implement a base Product class that represents general products, containing attributes like **name** and **price**.
    - Create derived classes such as Electronics and Clothing that **override** the base class method for calculating product discounts.
2. **Shopping Cart**:
    - Implement a ShoppingCart class that can hold a collection of products.
    - Overload the + operator to merge two shopping carts into one.
3. **Payment Processing**:
    - Implement a PaymentProcessor class that simulates **method overloading** to handle different payment methods (e.g., credit card and PayPal) using variable arguments.