Can you generate a classic java oop exercise that meets the following criteria: 1. level of difficulty: 7/10 2. should take about 2.5 hours to complete

**ChatGPT said:**

Sure! Here’s a **classic Java OOP exercise** that should take about **2.5 hours** to complete with a **difficulty level of 7/10**.

**Exercise: Library Management System (OOP)**

**Objective**

Design and implement a small **Library Management System** using **Object-Oriented Programming (OOP)** principles in Java.

**Requirements**

You must implement the following entities using proper **encapsulation, inheritance, and polymorphism**:

1. **Book** (Base class)
   * id (unique identifier, integer)
   * title (string)
   * author (string)
   * availableCopies (integer, represents the number of copies available)
   * borrow() method (reduces availableCopies by 1 if available, otherwise prints an error)
   * returnBook() method (increases availableCopies by 1)
2. **Member (Abstract Class)**
   * memberId (integer)
   * name (string)
   * borrowedBooks (List of borrowed books)
   * borrowBook(Book book) (abstract method)
   * returnBook(Book book) (removes book from borrowed list and increases available copies)
3. **RegularMember (Subclass of Member)**
   * Can borrow up to **3 books** at a time
   * Implements borrowBook(Book book)
4. **PremiumMember (Subclass of Member)**
   * Can borrow up to **5 books** at a time
   * Implements borrowBook(Book book)
5. **Library** (Manages book and member operations)
   * books (List of books available)
   * members (List of registered members)
   * addBook(Book book) (adds a book to the collection)
   * registerMember(Member member) (adds a member to the system)
   * borrowBook(int memberId, int bookId) (handles book borrowing)
   * returnBook(int memberId, int bookId) (handles book returning)

**Implementation Details**

* Use **constructors** to initialize objects.
* Use **lists** (ArrayList<Book>, ArrayList<Member>) to manage collections.
* Ensure that **borrowBook()** prevents exceeding the limit for members.
* Implement proper **toString()** methods for easy display of information.
* Write a **test class (LibraryTest)** with a main() method that:
  + Creates a library
  + Adds books
  + Registers members
  + Simulates borrowing and returning books

**Bonus (Optional)**

* Add a Librarian class with methods to search for books by title or author.
* Implement an OverdueNotification feature using a simple **HashMap<Member, List<Book>>**.

**Expected Outcome**

By the end of this exercise, you should have a functioning **Library Management System** where members can borrow and return books while respecting borrowing limits. The implementation should demonstrate **OOP principles like inheritance, encapsulation, and polymorphism**.

Would you like me to provide a starting template for the classes? 🚀