CS627B Lab 1

Encapsulation in Java

Emlyn Hegarty-Kelly

In your pair choose one of the 3 options below to complete.

Option 1: Personal Finance Tracker

Scenario:

You are tasked with building a basic personal finance tracker that allows users to record and categorize their daily expenses. The goal is to encapsulate data properly and provide controlled access through methods.

Requirements:

1. Transaction Class
- Private fields:
  - double amount
  - String category
  - LocalDate date
  - String description
- Public methods:
  - Constructor to initialize all fields
  - Getters and setters for each field
  - Validation in setters (e.g., amount must be positive)

2. BudgetTracker Class
- Private field:
  - List<Transaction> transactions
- Public methods:
  - void addTransaction(Transaction t)
  - double getTotalExpenses()
  - double getTotalByCategory(String category)
  - List<Transaction> getTransactions()

Advanced Features:

- Implement a method to remove or update a transaction
- Add a method to export a summary of expenses (e.g., to a formatted string)
- Add a method to get expenses within a date range

Option 2: Library Book Management System

Scenario:

A small library wants to manage its collection and track which books are checked out. You will create a system to manage books and their availability.

Requirements:

1. Book Class
- Private fields:
  - String title
  - String author
  - String ISBN
  - boolean isCheckedOut
- Public methods:
  - Constructor to initialize all fields
  - Getters and setters
  - Methods to check out and return books with validation

2. Library Class
- Private field:
  - List<Book> books
- Public methods:
  - void addBook(Book b)
  - void removeBook(String isbn)
  - Book searchByTitle(String title)
  - List<Book> searchByAuthor(String author)
  - List<Book> listAvailableBooks()

Advanced Features:

- Add a Member class with borrowing history
- Prevent a book from being checked out if already checked out

Option 3: Smart Home Device Controller

Scenario:

A smart home system needs to manage devices like lights, thermostats, and locks. You will create a system to register and control these devices.

Requirements:

1. SmartDevice Class
- Private fields:
  - String deviceName
  - boolean status (on/off)
  - String location
- Public methods:
  - Constructor
  - Getters and setters
  - Methods to turn on/off the device

2. SmartHomeController Class
- Private field:
  - List<SmartDevice> devices
- Public methods:
  - void registerDevice(SmartDevice d)
  - void toggleDevice(String deviceName)
  - List<SmartDevice> listDevices()

Advanced Features:

- Add subclasses like SmartLight, SmartThermostat with additional fields
- Add a method to simulate a "night mode" that turns off all lights and locks doors

Assignment Deliverables:

- Your mini project plan, in Txt format. Just explaining your overall thought process
- Java source files for the option your team choose. Each file should be commented explaining the code . (Please ensure names and student numbers are in each file)
- A Main class with a demo of the features

Assessment Criteria:

- Correctness: Code compiles and runs without errors; meets all functional requirements
- Encapsulation: Proper use of private fields and public getters/setters; appropriate validation
- Code Quality: Clear, readable, and well-structured code with meaningful names
- Advanced Features: Bonus for implementing advanced features effectively