CSE-214
Assignment on **Structural Design Pattern**

EduLearn Platform is an online education platform that has been successfully running for 2 years. They started with a simple structure where courses were a bit scattered. Now, they're expanding their business model and need to restructure their system to accommodate new organizational and pricing features.

Current System Architecture

Currently, the platform offers multiple **Courses** each having multiple **Lessons**.

- Each lesson has its duration.
- Based on the total duration of all lessons, the duration of the course is determined.

New Requirements

However, the platform wants to improve organization and introduce flexible pricing as given below:

1. Modular Organization

- Group related courses into Modules.
- Duration of Module will be the sum of the duration of all courses.
- Allow customers to purchase individual courses OR entire modules.
- Need to calculate the total price at any level (module or course).

2. Optional Add-ons

- Offer a Practice question set for each Module for $10.
  However, this is optional for the users.
- Offer Live mentor support for each Module for $20.
  This, too, is optional for the users.

3. Discount System

- **Multi-Module Discount**: $15 off when purchasing 2 or more modules.
- **Special Discount**: $12 off when purchasing item(s) with duration 5 hours or above (Could be multiple courses, single or multiple modules, etc.).
- **Developing Country Student Discount**: $10 off for students from developing countries.

Technical Goal

Currently, their code is a "spaghetti" mess of if-else statements.
Your goal is to implement the core engine of the platform to be flexible and organized using appropriate design patterns.

Key Requirements

1. **Unified Interface:** You must design the system so that the "Checkout" logic doesn't care what it is selling. Whether a student buys one single Lesson, one Course, or a giant Module, the system should call a single method (e.g., calculatePrice()) and get the correct total automatically.
2. **Multiple Discounts**: Besides, a person should be able to avail multiple discounts. A student from a developing country buying a module with a Practice Set should get both discounts and the add-on price added correctly.
3. **Transparency:** Finally, users should be able to see everything under a module with price and duration of each item.

Your implementation should—

- Demonstrate correct design pattern.
- Follow principles of software engineering.
- Be minimal and simple.
- Be written in Java.

Submission

Submit a ZIP archive containing the entire refactored **src/** directory. Ensure that your code compiles and runs correctly.

Distribution

Criterion	**Points**
Correctness & Functionality	20
Addressing Requirement 1	30
Addressing Requirement 2	30
Addressing Requirement 3	20
TOTAL	100