امیرحسین ایروانی‌منش 9931066 airavanimanesh@aut.ac.ir

تکلیف شماره ۵

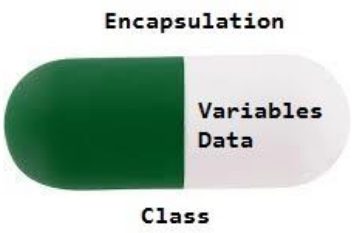
تمرین شماره ۱

الف)

Cohesion: مفهوم cohesion به تعداد و تنوع عملیات ها و اعمالی که توسط یک بخش خاص از برنامه انجام میشود (به طوری اختصاصی در جاوا، کلاس) اشاره میکند. وقتی یک کلاس اعمال متفاوت و متعددی را همزمان انجام دهد اصطلاحا آن یک کلاس low cohesion هست، ما برای تولید کلاس های ساختار یافته و در نتیجه آن برنامه با طراحی خوب، به دنبال high cohesion هستیم که یعنی یک کلاس به خصوص، مسئولیت تنها یک عمل منطقی را بر عهده داشته باشد. این اصطلاح را می توان علاوه بر کلاس، به متد و پکیج نیز تعمیم داد. از مزایای high cohesion می توان به فهم راحت تر یک بخش از برنامه، نام گذاری درست‌تر و واضح‌تر برای متغیر ها و فیلد ها و استفاده متعدد از همان متد یا کلاس و یا پکیج برای جلوگیری از طولانی تر شدن و از آن طرف باعث تولید کدی ساختار یافته تر می شود.

Coupling: مفهوم coupling به ارتباط چند بخش برنامه با هم و وابستگی آنها اشاره میکند، اگر دو بخش برنامه یا به طور اختصاصی در جاوا، کلاس های یک برنامه به شدت به هم مرتبط باشند و بهم وابسته باشند میگوییم آنها tightly coupled هستند. برای طراحی درست یک کلاس، هدف ما تولید کلاس های loose coupled که بهم به شدت وابسته نیستند، است. loose coupling به ما در فهم کد بدون نیاز به خواندن بخش های دیگر است و میتوان آن بخش از کد را بدون نیاز به تغییر بخش های دیگر، تغییر داد.

Encapsulation: مفهوم Encapsulation عبارت است از کپسوله سازی کد (منطق، رفتار و عملیات قابل اجرا توسط آبجکت) و داده ها و اطلاعات موجود در خود با هم در قالب یک بخش مانند یک کپسول که داروهای مختلف را به صورت ترکیبی در کنار هم شامل می شود که باعث می شود کاربر از کارکرد آن بخش از کد آگاهی نداشته باشد و فقط متغیر پاس بدهد و بگیرد.



کپسوله سازی یکی از ارکان برنامه نویسی شی گرا است که طی آن برنامه نویس دسترسی به برخی از اعضای کلاس از بیرون را آن طور و به آن اندازه که می خواهد تنظیم و مدیریت می کند. با استفاده از یکی از دو متدهای getter یا setter، می توان کلاس را به ترتیب فقط read-only یا write-only تعریف کرد. بدین وسیله می توان بر روی داده های کلاس کنترل بیشتری داشت.

Enumerated Type: Enumerated در لغت به معنای شمارش است و به طور کلی به داده هایی تعلق میگیرد که یک مقدار از پیش تعیین شده دارند و تغییر نمی کنند. مانند روزهای هفته یا فصل‌های یک سال که این مقادیر می توانند در متغیر های Enumerated

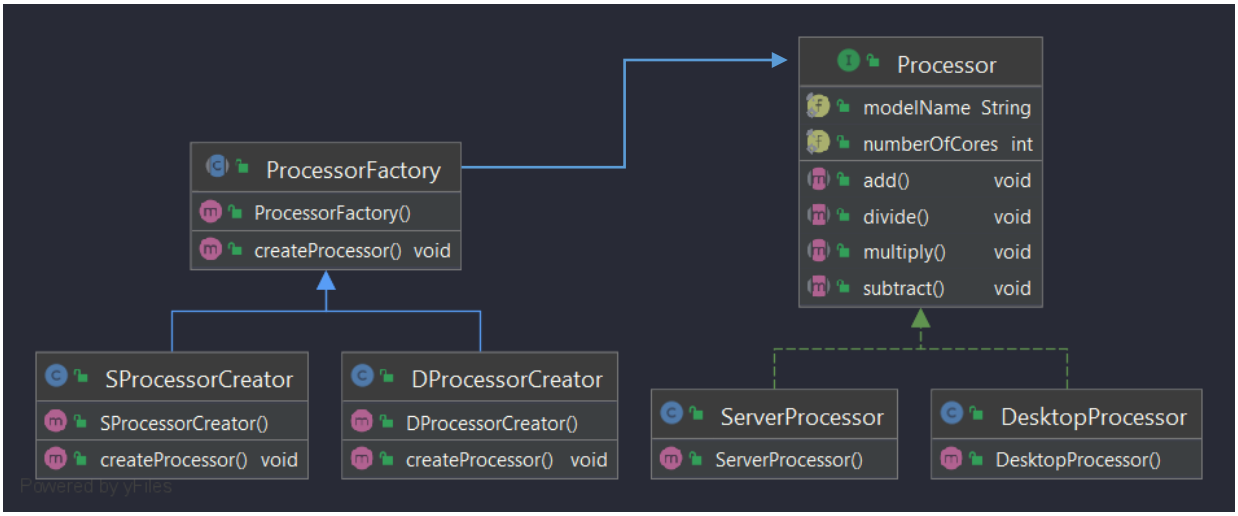
(که در جاوا از کلمه کلیدی enum استفاده می شود) ذخیره سازی کرد که تغییرپذیر نیستند و کاربری آسانی دارند. به زبان دیگر این نوع داده‌ها static و final هستند. این تغییر ناپذیری enum ها باعث می‌شود که داده‌های type-safe به شمار بیایند؛ یعنی راهی نیست که مقداری متفاوت با نوع آن به آن assign شود.

ب)

Factory Method: یک الگوی طراحی creational به شمار می آید که که درباره ساخت یک شئ می باشد. با استفاده از این الگوی طراحی می توان بر ساخت اشیاء کنترل بیشتر و بهتری داشت، نحوه کار این الگوی طراحی به این صورت است که با استفاده از interface، یک متد factory(نام فرضی هست) وجود داشته باشد و توسط ان متد، شئ های مورد نیاز ساخته می شوند. این الگوی طراحی از بهترین راه ها برای زمانی است که میخواهیم روند برنامه از کاربر مخفی بماند.

Adapter: این الگو مانند تبدیل ها در دنیای واقعی عمل میکنند. به گونه ای که کاربری شی ها را به هم مرتبط و متد های یکدیگر را بهم ترجمه می کند.به طور کلی دو نوع الگوی طراحی اداپتر وجود دارد که یکی object adapterو دیگری class adapter است. از آنجایی که class adapter نیاز به شی گرایی چند گانه دارد و این قابلیت در اکثر زبان های برنامه نویسی پشتیبانی نمی شود کم کاربرد است و ممکن است در اینده برنامه را دچار مشکل کند. از مزایای این الگوی طراحی این است که امکان استفاده مجدد و انعطاف پذیری کد فراهم می شود. و از معایب ان این است که گاهی برای ترجمه یک کلاس به کلاس دیگر نیاز به اداپتر های زنجیره وار داریم که برنامه را پیچیده میکند.

ج)



در این عکس، واضح است که یک کلاس فکتوری برای ساخت شئ های مورد نظر ساخته شده (Processor Factory) که دو زیر کلاس برای ساخت دو نوع پردازنده مذکور دارد.

تمرین شماره ۲

Library	
• Having Address field • Having Name field • Storing books in a list (HashSet for avoiding duplicate data) • Storing members in a List (HashSet for avoiding duplicate data) • print all books • print all members • Removing a member • Adding and removing a book • Storing list of borrows • Adding and removing a borrow from the list • Checking borrows to find Expired borrows to inform members that the deadline has come. • Finding a specific book	• Member • Book • Collection • Borrow • String

Member	
• Having field for name • Having field for email • Having field for subscription id • Having field for registration date • print details • storing borrows in a list • adding or removing borrows from list	• Borrow • String • Date • Collection

Book	
• Having a field for Name • Having a field for author • Print details (name and author)	• String

Borrow	
• having the member who borrowed the book. • having a field of borrow date and expiration date • print details	• Date • Member

همانطور که واضح است در بالای هر کارت نام کلاس و در سمت چپ مسئولیت ها (responsibility) و در سمت راست کارت همکاری ها وجود دارد. دلیل همکاری با Collection، در بعضی از کلاس ها، نیاز به ذخیره سازی داده ها در لیست است و همینطور همکاری با کلاس Date به دلیل نیاز به داشتن فیلد به فرمت تاریخ می باشد.