

تکلیف شماره ۴

تمرین شماره ۱

(الف)

Polymorphism: پلی مورفیسم یا چند ریختی به معنای همزمان چند فرم یا حالت داشتن است. برای مثال یک مرد که هم نقش کارمند و همسر و پدر را دارد. در جاوا دو نوع چند ریختی وجود دارد: ۱. چند ریختی زمان کامپایل ۲. چند ریختی زمان اجرای برنامه. برای ایجاد چند ریختی زمان کامپایل ما نیاز به **Overload** کردن متد ها داریم (اورلود کردن عملگر ها هم حساب میشود ولی چون جاوا چنین امکانی را فراهم میکند، ذکر نشده) و برای چند ریختی زمان اجرا نیز نیاز به **Override** کردن داریم.

Substitution: اصل جانشینی یکی از مهم ترین اصول برنامه نویسی شی گرا است. این اصل به ایجاد روابط درست بین سوپرکلاس ها و زیر کلاس ها و در نتیجه آن کد و برنامه ای **reusable** و **maintainable** داشته باشیم. در این اصل باید به اشتباهات رایجی که در آن اتفاق می افتد، توجه بسیاری کرد، برای مثال، اگر در یک برنامه یک کلاس پرنده داشته باشیم که یک متد **fly** داشته باشد، و یک زیرکلاس "مرغ" نیز داشته باشیم، از آنجایی که از لحاظ منطقی و بیولوژیکی مرغ یک زیرکلاس از پرنده است ولی در برنامه نویسی شی گرا اینطوری نیست، چرا که مرغ توانایی پرواز کردن را نداشته و در نتیجه متد **fly** نباید داشته باشد. و به عنوان یک مثال از کاربرد استفاده از آن، در یک برنامه رسم شکل، به جای اینکه برای هر شکل یک کلاس جداگانه داشت و هرکدام متد **draw** خاص خود را داشته باشند، میتوان از یک کلاس **shape** به عنوان سوپر کلاس استفاده کرد چون متد **draw** در بین همه آن ها مشترک است و بعد با استفاده از **polymorphism**، در صورت نیاز آن متد ها را **override** کرد.

Abstract Class: کلاس های **abstract** کلاس هایی هستند که با کلمه کلیدی **abstract** ساخته شده و در آنها میتوان متد های **abstract** که هیچ بدنه و دستوری داخل آنها وجود ندارد را ایجاد کرد. یک مثال استفاده از کلاس ابسترکت در ادامه میبینیم: ما یک سوپر کلاس **Animal** داریم که چند حیوان مختلف زیرکلاس آن هستند و از آن ارث بری میکنند، همه ی این حیوانات متد **sound** دارند پس چون مشترک هست در سوپرکلاس **Animal** قرار میدهیم ولی از آنجایی که صدای هر حیوان متفاوت از حیوان دیگر است از کلاس **abstract** استفاده میکنیم و **Animal** را با استفاده از کلمه کلیدی **abstract** به ابسترکت تبدیل کرده و در نتیجه متد **sound** آن را هم ابسترکت کرده ایم و بعد در هر کلاس هر حیوان آن متد را **override** میکنیم.

****قابل ذکر است که از یک کلاس ابسترکت نمیتوان یک شی ساخت.**

Interface: یکی دیگر از راه ها برای استفاده از انتزاع و **abstract** در جاوا، استفاده از **interface** است که در آن فقط متد های **abstract** میتوانند حضور داشته باشند. برای استفاده از متد های آن، این **interface** باید توسط دیگر کلاس ها **implement** شوند (به جای کلمه کلیدی **extends**).

(ب)

۱. غلط. در جاوا چنین عملی امکانپذیر نیست. و در صورت تلاش برای اجرای چنین عملی دچار ارور `cycling inheritance` میشویم.

۲. صحیح. همیشه در همه برنامه ها، اگر در کانستراکتور زیرکلاس، کلمه کلیدی `super` نوشته نشود، کامپایلر به صورت خودکار این عمل را انجام میدهد.

۳. صحیح. یک زیرکلاس تمامی اعضای سوپر کلاس را اعم از مقادیر، متد ها و `nested classes` را ارث بری میکنند. از آنجایی که کانستراکتور در این اعضا مشمول نمیشود، پس از آن ارث بری نمیشود.

۴. صحیح. در اورراید کردن متد ها، دسترسی به متد اورراید شده کمتر و یا حتی از بین می‌رود ولی در اورلود کردن متد دسترسی بین متد ها یکسان است.

۵. غلط. به دلیل اینکه کلاس فرزند(زیرکلاس) اصلاً به متد پرایوت سوپرکلاس و کلاس `parent` خود دسترسی ندارد، اورراید کردن آن متد امکانپذیر نیست.

۶. صحیح. یک کلاس فقط میتواند از یک کلاس ارثبری کند (با کلمه کلیدی `extends`) ولی میتواند از چند اینترفیس استفاده کند.

(ج) دو تفاوت عمده اورراید و اورلود کردن متد، نیاز داشتن به دو کلاس با رابطه ارث بری برای اورراید کردن است در صورتی که برای اورلود کردن متد وجود یک کلاس کفایت میکند. اورراید در زمان اجرای برنامه اتفاق می افتد ولی اورلود در پیش از زمان اجرای برنامه و در هنگام کامپایل اتفاق می افتد. پارامتر های ورودی را در اورلود کردن متد میشود تغییر داد ولی در اورراید باید پارامتر ورودی یکسان باشد. عمل اورلود را می توان بر متد های استاتیک نیز پیاده سازی نمود که چنین امکانی در `override` وجود ندارد.

تمرین شماره ۲

الف) ۱. مجاز. چون کلاس `husky` زیرکلاس `Animal` محسوب می شود.

۲. غیر مجاز. از آنجایی که `cow` زیرکلاس `mammal` محسوب می شود چنین چیزی امکان پذیر نیست ولی برعکس این دستور امکان پذیر است.

۳. مجاز. کلاس `bulldog` از کلاس `mammal` ارث بری کرده است پس مجاز است.

۴. غیر مجاز. به دلیل اینکه کلاس `parrot` زیرکلاس `bird` است چنین چیزی امکان پذیر نیست ولی برعکس این دستور امکان پذیر است.

ب) همه دستورات زیر مجاز هستند زیرا تمامی قوانین ارث بری و رعایت سلسله مراتب `dynamic type` و `static type` ها رعایت شده.