

گزارش کار پروژه عید - برنامه سازی پیشرفته (جاوا)

پاسخ سوالات قدم دوم

پیشوا آریز - 40313003

پاسخ سوال ۱

Shallow copy و Deep copy به مفاهیم مربوط به کپی کردن آبجکت ها برمی گردد. وقتی سعی می کنیم آبجکتی را کپی کنیم و آن را در جایی دیگر ذخیره کنیم باید در مورد این موضوع یعنی نوع کپی کردن آگاهی داشته باشیم. Shallow copy را با یک مثال بهتر متوجه می شویم:

فرض کنید کلاسی با نام Person داریم که در آن یک فیلد از نوع Address که آن هم خود یک کلاس است، وجود دارد.

```
class Address {
    public String city;
    public String country;

    public Address(String city, String country) {
        this.city = city;
        this.country = country;
    }
}

class Person {
    public String name;
    public Address address;

    public Person(String name, Address address) {
        this.name = name;
        this.address = address;
    }

    public Person copy() {
        return new Person(this.name, this.age, this.address);
    }
}
```

شاید این گونه به نظر برسد که متد copy می تواند آبجکتی جدید از نوع Person بازگرداند و مشکلی در پیاده سازی آن وجود ندارد. در حالی که با تست کردن عملکرد آن می توان متوجه شد که مشکل آن چیست. حال با ساختن یک آبجکت از نوع Person و کپی کردن آن در آبجکتی دیگر با استفاده از متد copy کدی مانند زیر خواهیم داشت:

```
public class Main {
    Address location = new Address("Hewler", "Kurdistan");
    Person person1 = new Person("Hozan", location);
    Person person2 = person1.copy();

    person2.name = "Dilan";
    person2.address.city = "Amed";

    System.out.println(person1.name + " is from " + person1.address.city);
    System.out.println(person2.name + " is from " + person2.address.city);
}
```

با مشاهده خروجی این کد، می توان فهمید تغییری که در یکی از فیلدهای person2 انجام شد، باعث همین تغییر در person1 نیز شده است. البته این فیلد از نوع Address بوده و به دلیل این که رفرنس آن از person1 به person2 کپی شده است، تغییرات مشابهی داشته اند؛ وگرنه تغییری که در person2.name اعمال شده است، موجب تغییر در person1.name نشده است، چون فیلد name از نوع primitive است. لذا آبجکت ها و رفرنس های داخلی هر آبجکت همان گونه باقی مانده اند و نسخه های جدیدی از آن ها به وجود نیامده است. این نوع کپی کردن که در آن آبجکت کپی شده، با همان رفرنس های داخلی قبلی کپی می شود، Shallow copy می باشد.

این موضوع در این جا، با توجه به مشکلی که در پایان قدم اول پروژه مشاهده شد، در اصل نوعی عملکرد نامطلوب دیتابیس است، لازم است تا متد مربوط به کپی کردن آبجکت ها به گونه ای صحیح پیاده سازی شود تا رفرنس هایی که در آبجکت ها مورد استفاده قرار گرفته است، متمایز از هم باشند و به یک فضای واحد در حافظه اشاره نکنند.

با این توصیف، می توان پیش بینی کرد که Deep copy چه تمایزاتی دارد. Deep copy به نوعی از کپی کردن آبجکت ها اشاره می کند که در آن، از رفرنس ها و آبجکت های داخلی هم کپی گرفته می شود و به جای هر کدام از آن ها، نسخه های جدیدی در آبجکت کپی شده قرار می گیرند. برای این منظور می توان پیاده سازی متد copy را به شیوه زیر تغییر داد:

```
public Person copy() {
    Person copied = (Person) super.clone();
    copied.address = new Address(this.address.city, this.address.country);
    return copied;
}
```

همان گونه که مشاهده می شود، نسخه جدیدی از فیلد address که از نوع reference می باشد، ایجاد شده است. با این نوع پیاده سازی، دیگر هرگونه تغییر در هر یک از فیلدهای آبجکت های person1 یا person2، تاثیری بر دیگری نمی گذارد.

پاسخ سوال ۲

در اینجا به خاطر یکی از قابلیت های جاوا به نام Covariant return types، می توان نوع خروجی از متدها را تغییر داد، به شرطی که موارد زیر برقرار باشند:

۱. نوع خروجی از متد override کننده، باید یک زیرکلاس از نوع خروجی متد override شده باشد.

۲. این تغییر سیگنچر فقط برای reference type ها پاسخگو است و برای primitive type ها امکان پذیر نمی باشد.

با توجه به موارد بالا، چون کلاس Human زیرکلاسی از کلاس Entity است، می توان نوع خروجی متد copy را از Entity به Human تغییر داد.

پاسخ سوال ۳

```
shadow@ROG-Zephyrus ~/DEV/uni/ap/todo-list <clone-method>  
└─ j Main  
ali's name in the database: Ali
```

اسکرین شات از خروجی کد حین اجرای بخش تست کد

پاسخ سوال ۴

متد clone در جاوا برای ایجاد کپی از یک آبجکت استفاده می‌شود. این متد در کلاس Object تعریف شده است. برای استفاده از آن شرایطی مانند implement کردن اینترفیس Cloneable توسط کلاس مورد نظر (برای جلوگیری از اکسپشن مربوط) و override کردن متد clone و فراخوانی super.clone() وجود دارد. استفاده از این متد به تنهایی، آبجکت مورد نظر را صرفاً Shallow copy می‌کند و برای پیاده‌سازی Deep copy باید به صورت دستی رفرنس‌های موجود در داخل آبجکت‌ها را کپی کرد.

پاسخ سوال ۵

مطابق سورس کد خود جاوا، اینترفیس Cloneable هیچ متدی در تعریف خود ندارد و استفاده از آن برای کلاسی که متد clone را پیاده‌سازی می‌کند صرفاً جنبهٔ مارکر دارد و نشان می‌دهد که کلاس ذکر شده از cloning پشتیبانی می‌کند. از طرفی ضرورت استفاده از این اینترفیس می‌تواند به مفاهیم clean code به‌خصوص واضح‌تر شدن و جلوگیری از عملکرد غیرمنتظره و نامطلوب مربوط شود.