برنامه نویسی پیشرفته

زمستان و بهار ۹۹-۱۳۹۸ - دانشکده علوم ریاضی

دانشگاه صنعتی شریف

با توجه به شرایط خاص پیش آمده تیم درس برنامه نویسی پیشرفته تصمیم گرفتند که یک پرسش‌نامه بدون تاثیر در ارزیابی برای بررسی میزان پیشرفت مطالعه برگزار کند. هدف از این پرسش‌نامه بررسی پیشرفت عملکرد و مطالعه شما است.

هدف از این پرسش‌نامه این موارد است:

- جبران فاصله ایجاد شده میاد دانشجوها با همدیگر که امکان ارائه بازخورد پیشرفت مطالعه به یکدیگر را ایجاد می‌کند.
- جبران فاصله ایجاد شده میان دانشجوها و تیم درس برای دریافت بازخورد پیشرفت تحصیلی
- بازخورد هر دانشجو به خود در رابطه با پیشرفت مناسب در مطالعه و یادگیری مفاهیم از طریق منابع درس
- جهت‌دهی به اشکالاتی که شما ممکن هست هنوز در جریان وجود نقطه ضعف خود در این رابطه نباشید و رفع آنها در جلسه‌های رفع اشکال آنلاین
- دقت کنید که این پرسش‌نامه تنها مرجع برای بازخورد پیشرفت مناسب شما در مطالعه منابع نیست. تمرین‌ها و پروژه نیز سهم بزرگی در این مساله دارند. پس حتما پس از پاسخ به سوال‌ها و ارسال پاسخ‌ها، اشکال‌ها و ابهام‌هایی که داشتید در جلسه‌های آنلاین رفع اشکال در میان بگذارید و رفع کنید و از این فرصت استفاده کنید.

توضیحات

- نتیجه این پرسش‌نامه تاثیری در ارزیابی نهایی این درس ندارد.
- این پرسش‌نامه برای اطمینان بیشتر از اینکه مسیر درس را درست طی می‌کنید طراحی شده.
- اگر نیاز به بررسی صحت پیشرفتتان دارید حتما در این پرسش‌نامه شرکت کنید.
- در صورتی که با مطالب درس به درستی پیش آمده باشید می‌توانید به تمام سوال‌ها پاسخ دهید.
- در صورتی که به بخشی از هر سوال تسلط ندارید یا احتیاج به بررسی صحت پاسخ‌ها دارید حتما در جلسه‌های رفع اشکال شرکت کنید و اشکال یا ابهام‌های خود را رفع کنید.
- سعی کنید جواب‌ها کوتاه و دقیق باشند که مرور جواب در جلسه رفع اشکال سریع‌تر انجام شود.
- از آنجایی که این پرسشنامه برای یادگیری طراحی شده می‌توانید در پر کردن سوال‌ها با هر فردی مشورت و همفکری کنید.

نحوه انجام پرسش‌نامه

- برای پاسخ به این پرسش‌نامه یک نسخه از این فایل را از منو فایل و گزینه گرفتن یک کپی برای خود ایجاد کنید و جواب‌های آن را در همین فایل بنویسید.
- پس از جواب دادن به سوال‌ها آن را در قالب PDF دانلود کنید.
- فایل PDF در یک ریپازیتوری github بارگذاری کنید.
- آدرس این ریپازیتوری را در یک فایل یک خطی با پسوند جاوا داخل کوئرا و در بخش پرسشنامه بررسی پیشرفت بارگذاری کنید.

سوال‌ها

سوال ۱

خروجی این برنامه را بدست بیاورید و به ازای هر خط توضیح دهید که چرا به این خروجی رسید؟

```
class Classes {
   static class A {
       static int intValue = 0;
       int integerValue = 20;

       A() {
           integerValue = 5;
           printValue();
           print();
       }

       void printCaller() {
           print();
       }

       void printValue() {
           System.out.println("B:" + integerValue);
       }

       void print() {
           System.out.println("A:" + intValue);
       }
   }

   static class B extends A {
       B(int v) {
           intValue = v;
           integerValue = 15;
           printValue();
           print();
       }

       void print() {
           System.out.println("B:" + intValue);
       }

       void printSuper() {
           super.print();
       }

       void printCaller() {
           printValue();
           super.printValue();
       }

       void printValue() {
           System.out.println("B:" + integerValue);
```

```
            super.printValue();
        }
    }

    static public class C extends A {
        void printCaller() {
            System.out.println("B:" + integerValue);
        }

        void print() {
            System.out.println("A:" + intValue);
            super.printCaller();
        }
    }
}

class Problem1 {
    public static void incrementValue(Classes.A object) {
        object.intValue++;
        object.integerValue++;
    }

    public static void incrementValue(int firstValue, int secondValue) {
        firstValue++;
        secondValue++;
    }
    public static void main(String[] args) {
        Classes.A a = new Classes.A();
        Classes.B b = new Classes.B(10);
        Classes.A c = b;

        b.print();
        c.print();
        ((Classes.A) b).print();
        b.printSuper();
        a.printCaller();
        b.printCaller();
        c.printCaller();
        incrementValue(a);
        a.printCaller();
        incrementValue(b);
        b.printCaller();
        incrementValue(c);
        c.printCaller();
        incrementValue(b.intValue, b.integerValue);
        b.printCaller();
        c.printCaller();
    }
}
```

پاسخ:
با اجرای برنامه‌ی main، ابتدا یک اینستنس از کلاس درونی A در کلاس Classes ساخته می‌شود. در سازنده فراخوانده شده، تابع printValue فراخوانده می‌شود و عبارت B:5 چاپ می‌شود و سپس تابع print که مقدار متغیر استاتیک را چاپ میکند :0A

سپس یک نمونه از کلاس B، که از A ارث می‌برد، اینستنس گرفته می‌شود. در سازنده فراخوانده شده، با توجه به اینکه متدهای فراخوانده شده همگی در کلاس B، override شده‌اند، توابع مذکور از کلاس B صدا زده می‌شوند. ابتدا printvalue از کلاس B فراخوانده می‌شود که ابتدا B:5 را چاپ می‌کند و سپس همین تابع را در A فرامی‌خواند که آن هم همین خط را چاپ می‌کند. سپس دستور print در سازنده A فراخوانده می‌شود که چون دوباره در B override شده است، متد از B فراخوانده می‌شود و لذا B چاپ می‌شود اما چون از کلاس A فراخوانده شده مقدار intValue همچنان 0 است. پس از این سازنده B فراخوانده می‌شود و B:15 چاپ می‌شود و سپس printvalue کلاس پدر فراخوانده می‌شود اما توجه شود که چون integervalue که در سازنده B مقداردهی شد، در خود کلاس B متغیری با این اسم موجود نبود، متغیر کلاس A تغییر داده می‌شود و لذا خروجی super.printvalue نیز همان B:15 است. سپس print فراخوانده میشود و به سادگی B:10 چاپ می‌شود.
پس از این خط سوم کلاس main اجرا می‌شود. این خط درواقع یک upcast است.
دستور print از b فراخوانده می‌شود.
دستور print که توسط c فراخوانده می‌شود، دستور print را در کلاس B فرا می‌خواند.
در این خط نیز دوباره دستور print از B فراخوانده می‌شود.
b.superprint نیز بدیهی اجرا می‌شود
a.printcaller نیز با توجه به اینکه متغیر intvalue از نوع استاتیک است، همان مقدار که در b داده شده است را دارد و A:10 را چاپ می‌کند.
خروجی دستورات b.printcaller و c.printcaller دقیقا یکی است و هر یک 3 مورد B:15 چاپ می‌کنند که توضیح آن مشابه توضیح ارائه شده در قسمت سازنده B است.

خط بعدی با فراخواندن incrementvalue بر روی شی، مقادیر دو متغیر در a یک واحد افزایش می‌یابند و یک خط در خروجی چاپ می‌شود.

مشابها دستور incrementvalue روی b و c اجرا می‌شوند و در هر مرحله متغیرهای موجود یک واحد افزایش می‌یابند و در چاپ‌ها انجام می‌شوند.
دستور incrementValue که با 2 متغیر فراخوانده شده است، چون تغییرات انجام شده بازگردانده نشده‌اند، هیچ تغییری اعمال نشده و دو خط بعدی که چاپ می‌کنند، دقیقا یکسان هستند و برابر خروجی چاپ قبلی که از c فراخوانده شد، است.

سوال ۲

توضیح دهید که هدف از ارث بری در شی گرایی چیست. چه زمان از composition و چه زمان از inheritance استفاده می‌کنیم؟ چگونه می‌توانیم از سازنده پدر را فراخوانی کنیم؟ چگونه می‌توانیم سازنده دیگری از خود کلاس را فراخوانی کنیم؟

پاسخ: code reuse یکی از اهدافشه. دید ساختاری به اجسام و به کد. خوانا کردن کد
با توجه به نوعی محدودیت در تنها 1 کلاس برای ارث بردن، inheritance به تنهایی پاسخ‌گوی نیازها نیست. از طرفی هم لزومی نیست که یک کلاس از یک کلاس دیگر لزوما ارث ببرد بلکه می‌تواند با داشتن نمونه‌ای از آن در خود، بدون ارث بردن و با استفاده از ترکیب، نیاز خود را برطرف کند.
با استفاده از کلیدواژه ()super در ابتدای سازنده کلاس فرزند، سازنده پدر فراخوانی می‌شود.
با استفاده از کلیدواژه ()this.

سوال ۳

توضیح دهید که چرا از رابط‌ها (interface) استفاده می‌کنیم. چه محدودیت‌هایی نسبت به یک کلاس دارند و چرا امکان پیاده‌سازی متد در آنها داده شده است؟

Interface ها در فراخوانی توابعی که بین کلاس‌ها مشترک هستند مناسب هستند و در بعضی استفاده‌ها، به نوعی واجب هستند. فرض کنید که مجموعه کلاس‌های کارت‌های مختلف بازی را نوشته‌ایم. عملی مانند throw card عملی است که در بین تمامی انواع کارت‌ها مشترک است و ما برای کنترل بازی دیگر لازم نیست که اینستنی از همان کارد را در همان نوع رفرنس ذخیره کنیم و می‌توانیم همه‌ی کارت‌ها را در آرایه‌ای از card interface قرار دهیم و اکنون به سادگی می‌توانیم روی این آرایه سوییپ کرده و متد مشترک throw card را فرا بخوانیم.
Interface ها در واقع شمای کلی از توابعی که یک کلاس که آن را implement می‌کند را ارائه میکنند و هر کلاسی که آن را implement می‌کند ممکن است به شکل دلخواهی آن را پیاده کند.
از interface ها نمی‌توان اینستنس گرفت. متغیرهای داخل interface همگی final و public هستند.
خب داده شده دیگه!

سوال ۴

کلاس انتزاعی (abstract) چیست و چه زمانی در مدل‌سازی از یک کلاس انتزاعی استفاده می‌کنیم؟ این نوع کلاس چه تفاوتی با رابط(interface) دارد؟
نوعی کلاس که قابل نمونه گیری نیست. اصولا تمامی کاربردهای interface برای این گونه کلاس‌ها نیز ممکن است و بیشتر چرا که بر خلاف interface که همه چیز public بوده و تمامی متغیرها هم final هستند، در کلاس ابسترکت می‌توان نوع توابع و کلاس‌ها را متفاوت گرفت و مزومی بر پابلیک بودن آن‌ها نیست.

سوال ۵

override کردن تابع و متغیر چه تاثیری در عملکرد متد در یک کلاس فرزند می‌گذارد؟ چطور می‌توانیم پس از override شدن یک متد در کلاس فرزند در هر کدام از مکان‌های زیر به نسخه هم نام آن متد در کلاس پدر دسترسی پیدا کنیم؟

- متدی داخل کلاس پدر: با استفاده از کلیدواژه super
- متدی داخل کلاس فرزند: در این حالت در واقع overload صورت گرفته و با استفاده همان نام خود تابع و تنها با دادن درست ورودی‌های تابع، متد هم‌نام دیگر فراخوانی می‌شود.
- خارج از دو کلاس: با طی درست آدرس آبجکت و یا ایمپورت کردن کلاس متناظر آن متد.

سوال ۶

توضیح دهید که منظور از چندریختی در شی گرایی چیست و چه مزیتی ایجاد می‌کند.
این که می‌توان هنگام اینستنس گرفتن از یک کلاس، آن را داخل یک سوپرکلاس و یا حتی یک interface که در مرحله توسط این کلاس یا سوپرکلاس‌های آن implement شده، قرار دهیم. به نوعی مثالی که در سوال 3 و interface توضیح دادم را در اجماع با این قسمت بایستی به کار برد.

سوال ۷

چرا از توابع و متدها در زبان برنامه نویسی استفاده می‌کنیم؟ در طراحی برنامه و شکستن آن به توابع و متدهای مختلف چه نکته‌هایی را باید رعایت کرد که خوانایی آن بیشتر شود و پیچیدگی اضافی نداشته باشیم؟
code reuse ؟؟؟.

سوال ۸

کلاس درونی (inner class) چه انواعی دارد و هر کدام چه کاربردی در مدل‌سازی و توصیف موجودات دارد؟ چگونه می‌توانیم یک شی از هر نوع ایجاد کنیم؟ در صورت override شدن یک متد یا متغیر توسط یک کلاس درونی چگونه می‌توان به نسخه override شده از کلاس بیرونی دسترسی پیدا کرد؟

استاتیک و غیر استاتیک. یکی از فرق‌های این دو در این است که در حالت استاتیک نیازی به داشتن instance از نوع کلاس بیرونی نیست و کلاس داخلی به تمامی متغیرها و متدهای استاتیک کلاس بیرونی دسترسی دارد اما در حالتی که کلاس داخلی غیر استاتیک باشد، برای به اینستنس گرفتن ابتدا بایستی یک نمونه از کلاس بیرونی موجود باشد تا سپس بتوان از کلاس داخلی نمونه گرفت. در حالت غیر استاتیک البته کلاس داخلی به تمامی متغیرها و روش‌های کلاس بیرونی، حتی آن‌هایی که پرایوت هستند نیز دسترسی دار.

سوال ۹

کلمه کلیدی final روی هر کدام از موارد زیر چه تاثیری دارد؟

- تابع و متد: این توابع توسط فرزندان این کلاس قابل override نیستند.
- تعریف کلاس: این کلاس‌ها قابل extend نیستند.
- یک متغیر از نوع شی: آدرس متغیرها کانستنت هست ولی می‌توان محتویات درونی آن‌ها را تغییر داد.
- یک متغیر از نوع پایه: این متغیرها کانستنت هستند.

سوال ۱۰

کلمه کلیدی static روی هر کدام از موارد زیر چه تاثیری دارد؟

- تابع و متد: نیازی به اینتستن از کلاسی که این توابع داخل آن نوشته شده‌اند نیست و می‌توان این توابع را بدون داشتن نمونه فراخوانی کرد.
- تعریف کلاس: کلاس استاتیک تنها می‌تواند یک کلاس درونی باشد. در این صورت می‌توان کلاس درونی را بدون داشتن نمونه‌ای از کلاس بیرونی اینستنس گرفت.
- یک متغیر از نوع شی: فک کنم مثل متغیر نوع پایه استاتیک.
- یک متغیر از نوع پایه: این متغیرها بدون وجود اینستنس از کلاسی که در آن قرار دارند قابل فراخوانی هستند و مقدار اولیه داشته و ...