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();
    }
}
```

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؟

در طراحی ممکن است اشیائی وجود داشته باشد که خود یک مفهوم از یک کلاس کلی باشند و رابطه IS-a میان آن ها وجود داشته باشد. اولا برای طراحی بهتر و ثانیا برای جلوگیری از تکرار (تکرار چیزهای مشابه به عنوان یک عامل بد در برنامه نویسی شیء گرا شناخته می شود.) از ارث بری استفاده می کنیم. به طور مثال قصد پیاده سازی یک باغ وحش را داریم در این باغ وحش حیوانات مختلف از جمله شیر، مار، لاک پشت و کرم وجود دارد. فرض کنید ابرکلاس حیوان وجود دارد که خود به دو زیر کلاس مهره دار و بی مهره تقسیم می شود و از طرفی مهره داران به شیر، مار و لاک پشت تقسیم می شوند. حال یک سری ویژگی و رفتار مشترک بین تمامی حیوانات وجود دارد که کافیست در کلاس حیوان پیاده سازی یا تعریف (ممکن است رفتار مثلا خوردن یکسان باشد ولی نحوه پیاده سازی برای وارثان متفاوت باشد.) شوند و از طریق ارث به زیر کلاس های آن برسند و دیگر نیاز به باز پیاده سازی آن نباشد.

هر زمان رابطه has-a بین دو چیز برقرار بود از Composition استفاده می کنیم مثلا دانشکده استاد دارد.
هر زمان رابطه Is-a بین دو چیز برقرار بود از Inheritance استفاده می کنیم مثلا شیر یک حیوان است.

با دستور super به صورت Super(Father constructor parameter) می توان Constructor پدر را فراخوان کرد.
برای اشاره به متدی که در پدر قرار دارد نیز می توان به صورت super.method استفاده کرد.

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؟

رابط ها درگیر پیاده سازی جزئیات نمی شوند و تنها کلیات رفتار را نشان می دهند پس از آن ها به عنوان توصیف طراحی استفاده می کنیم.
رابط ها در ارث بری و چند ریختی آزادی عمل بیشتری به ما می دهد .
خوانایی کد و پی بردن به منطق کد را بهتر می کند.
می توان اول رفتارهایی که به نظر نیاز می آید با رابط مشخص کرد و بعد از آن به پیاده سازی پرداخت.

از یک رابط نمی توان شیء ساخت و از طرفی متد های آن به صورت ضمنی Public و abstract هستند.

در رابط ها مگر می توان پیاده سازی متد داشت ؟؟؟؟

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؟

کلاس های انتزاعی کلاس هایی هستند که نمی توان از آنان مستقیما شیء ساخت.

زمانی که بخواهیم حداقل یک متد انتزاعی تعریف کنیم. وقتی بخواهیم رفتار ها را بیان کنیم بدون آنکه پیاده سازی آن را داشته باشیم و بعد در فرزندان آن را پیاده سازی کنیم. مثلا حیوانات غذا می خورند ولی غذا خوردن شیر با ماهی متفاوت است پس می توان گفت که حیوان یک مفهوم انتزاعی است که رفتار انتزاعی غذا خوردن دارد ولی نحوه ی هر غذا خوردن هر نوع حیوان متفاوت و در کلاس آن پیاده سازی می شود.

در رابط باید تمامی متد ها انتزاعی باشند ولی در کلاس انتزاعی می توان متدی غیر انتزاعی هم داشت.

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
- متدی داخل کلاس فرزند
- خارج از دو کلاس

باعث گسترش آن نسبت به پدر می شود مثلا متد جابه جایی در تمرین مار و پله در بخش نردبان بازکن را جابه جا می کند ولی در نردبان خوب علاوه بر جابه جا کردن که از نردبان به ارث می رسد با بازتعریف می توان ویژگی افزایش امتیاز را نیز به این متد در فرزند اضافه نمود.
دسترسی به متدی در داخل پدر زمانی که در فرزند هستیم: super.method
زمانی که متدی داخل فرزند باشد و در همان بخواهیم از آن استفاده کنیم: This
برای استفاده از متد در خارج با ساختن یک شیء از کلاس مد نظر یا اگر متد ما Static باشد با صدا زدن نام کلاس

سوال ۶

توضیح دهید که منظور از چندریختی در شی گرایی چیست و چه مزیتی ایجاد می‌کند.

وقتی متغیر را از نوع بالاتر (پدر) بسازیم و با new نوع جزئی آن را مشخص کنیم از چند ریختی استفاده کردیم.
به طور مثال:

```
Animal a = new Line();
```

زمانی که یک رفتار مشترک از اشیاء که پدر یکسانی دارن قرار است برای تمامی اشیاء فراخوانی شوند نیازی نیست تا برای هر نوع (فرزند) از اشیاء یک حلقه نوشت بلکه با نوشتن حلقه روی پدر (در صورتی که اشیاء به صورت چند ریختی تعریف شده باشند.) تمامی اشیاء فراخوانی می شوند و خود کامپایلر بر اساس نوع new تشخیص می دهد کدام متد را از فرزندان پیاده سازی کند.

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؟
در برنامه نویسی شیء گرا سه مشخصه ویژگی و حالت/ متد و رفتار/ هویت مطرح است. از توابع برای نشان دادن یا انجام دادن یک رفتار یا انجام یک کار استفاده می شود.

نام گذاری توابع مناسب باشد گاهی با گذاشتن annotation ها می توان خوانایی را افزایش داد. برای طراحی می توان اول از رابط و مفاهیم انتزاعی کمک گرفت سپس به پیاده سازی توابع بپردازیم.

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؟

کلاس های داخلی می تواند public, protected, package access, private و final و static و بی نام با oterclass.this

سوال ۹

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

- تابع و متد

نمی توان آن را override کرد یا برای زیر کلاس ها مخفی است.

- تعریف کلاس

نمی تواند یک زیر کلاس داشته باشد و از آن ارث بری داشته باشیم

- یک متغیر از نوع شی

این متغیر دیگر تغییر نمی کند و ورودی آن از طریق سازنده است.

- یک متغیر از نوع پایه

تنها یکبار می تواند مقدار دهی اولیه شود.

سوال ۱۰

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

- تابع و متد

نیازی به ساخت شیء از کلاس نیست و با استاتیک کردن یک متد می توان آن را با نوع کلاس استفاده کرد یعنی آن متعلق به خود کلاس است نه اشیاء آن

- تعریف کلاس

زمانی که یک کلاس درون یک کلاس دیگر باشد می تواند از static استفاده کند و نمی توان از آن شیء ساخت و به اعضای غیر static در دسترسی نمی دهد.

- یک متغیر از نوع شی

یک شیء متعلق به کلاس می شود و از طریق خود کلاس فراخوانده می شود و دسترسی به آن داده می شود.

- یک متغیر از نوع پایه

یک متغیر متعلق به کلاس می شود و اشیاء به آن دسترسی ندارند و از طریق خود کلاس فراخوانده می شود و دسترسی به آن داده می شود.