نکات کلیدی و بخش های اضافه شده به تمرین چهار

با توجه به مباحث مطرح شده در تمرین ۲ و ۴، به نکات زیر دقت کنید.

1. پیاده سازی: کلاس درس کارشناسی و کارشناسی ارشد باید کاملا جدا از هم بوده و با استفاده از مفاهیم ارث بری پیاده سازی شوند. تکرار ویژگی های پدر(کلاس درس) در هر یک از این کلاس ها نمره **منفی** دارد! استفاده از مفاهیم شیء گرایی، تفاوت در Private یا Public بودن و استفاده از Static در موارد مورد نیاز باید کاملا رعایت شود. شرط پیاده سازی getter و setter هم چنان پابرجاست. جاهایی که استفاده از چند ریختی (polymorphism) امکان پذیر است باید با این روش نوشته شود.

اجرا: ادمین از طریق منو(که شما طراحی کرده اید) می تواند یک (یا چند) درس را اضافه کند. وظیفه شما این است که کانستراکتور های دو کلاس ذکر شده را جوری طراحی کنید که به شکل خودکار بفهمد که باید شیء از نوع درس کارشناسی باشد یا ارشد و به شکل کانستراکتور های اوورلود شده باشند (با توجه به متغیر ها نوع شیء خود به خود تشخیص داده می شود)! تمامی ویژگی هایی که باید وجود داشته باشند مانند :

- ID (یک عدد 2 رقمی یونیک – مخصوص کلاس فرزند)
- نام درس (مثلا ریاضی ۲)
- واحد درس (۱ یا ۲ یا ۳)
- زمان کلاس (بین 1 تا ۸ از روز های 1 تا ۵)
- ظرفیت درس (که همواره باید بروز باشد)
- ورودی سال مجاز(مثلا ۹۷)
- رشته مربوط به درس (مثلا مهندسی کامپیوتر که یک شیء از نوع کلاس رشته می باشد و **پس از اضافه شدن درس به یکی از دو لیست دروس اجباری یا اختیاری رشته مقدار می گیرد)**
- درس پیش نیاز(یک درس کافی می باشد و **باید از قبل تعریف شده باشد** و شیء مربوط به درس پیش نیاز به عنوان این ویژگی برای درس جدید ست شود. می تواند این مقدار برای یک درس Null باشد. به این معنی که پیش نیاز ندارد و باید در مانستراکتور این نکته رعایت شود – مخصوص کلاس فرزند)
- لیست دانشجویان (که لیستی از جنس شیء دانشجوی کارشناسی استعداد درخشان یا کارشناسی عادی یا کارشناسی ارشد است و قبل انتخاب واحد مقداری ندارد. توجه داشته باشید که کلاس دانشجوی کارشناسی و ارشد، فرزند کلاس دانشجو و دانشجوی استعداد درخشان و عادی فرزند کلاس دانشجوی کارشناسی هستند)
- استاد درس (شیء از نوع استاد)
- ماکزیمم ظرفیت (عددی **ثابت** و متفاوت برای کارشناسی و ارشد می باشد – مخصوص کلاس فرزند)
- دروس پیشنیاز مخصوص ارشد (لیستی از شیء درس که از نوع کارشناسی می باشد – مخصوص کلاس فرزند کارشناسی ارشد)

حالا پس از اینکه این درس ها ساخته شدند باید قابلیت نمایش همه آنها وجود داشته باشد. همینطور گزینه جدایی برای نمایش اطلاعات دانشجویان یک درس خاص و اطلاعات استاد یک درس خاص باید وجود داشته باشد (این موارد جزء گزینه های منوی مربوط به نمایش اطلاعات کلاس ها که شما طراحی می کنید است). برای مثال منو دارای این ترتیب باشد: منوی اصی > 2.اطلاعات دروس > 1. اطلاعات دانشجویان درس > نهایتا ی وارد کردن ID درس اطلاعات دانشجویانی که درس را اخذ کرده اند نمایش داده می شود.

قسمت اضافه شده: اطلاعات تمامی درس ها بلافاصله بعد از ساخته شدن شیء باید در فایل Subjects.txt ذخیره و هنگام نمایش هر گونه اطلاعات مربوط به درس، از همان فایل خوانده شود. در این بخش استفاده از آبجکت های ساخته شده اشتباه است چرا که با اجرای دوباره برنامه این شیء ها قطعا وجود نخواهند داشت. **هدف ذخیره سازی اطلاعات گذشته و بازیابی آنها می باشد!**

نمره اضافی:

- جداسازی ویژگی های زمان کلاس و ظرفیت درس با اضافه کردن کلاسی از جنس کلاس (منظوری همون کلاسی که میز هست توش و استاد میاد تدریس میکنه) و اضافه کردن ویژگی درس از نوع شیء درس به کلاس کلاس (اووف سخت بود توضیحش). به **نکته 9** حتما توجه شود.
- دریافت و نمایش اطلاعات توسط GUI

توجه کنید: کانستراکتور ها ی کلاس با توجه به تعاریف بالا و صورت پروژه 2، 2 حالت کلی دارند: با ID و نام درس، با تمامی اطلاعات. متد های محاسبه ظرفیت خالی و یا متد افزایش ظرفیت درس(یا کلاس که حتی با فرض افزایش یافتن نباید از ماکزیمم ظرفیت عبور کند. مثلا ظرفیت ماکزیمم 25 است. ظرفیت اولیه انتخاب شده 15 می باشد. حالا 3 دانشجو درس را اخذ کرده اند و ظرفیت خالی برابر با 12 می باشد. شرط افزایش ظرفیت به این شکل است: تعداد دانشجویان - ظرفیت خالی - 25 => مقدار افزایش ظرفیت که در این مثال برابر 10 => مقدار افزایش ظرفیت می باشد) باید هنگام انتخاب واحد به درستی کار خود را انجام دهند.

2. پیاده سازی: کلاس دانشجوی کارشناسی و کارشناسی ارشد باید کاملا جدا از هم بوده و با استفاده از مفاهیم ارث بری پیاده سازی شوند. همینطور کلاس دانشجوی استعداد درخشان و عادی در مورد کلاس پدر خود(کلاس دانشجوی کارشناسی). تکرار ویژگی های پدر(کلاس دانشجو و کلاس دانشجوی کارشناسی(برای کلاس های فرزند سطح ۲ یعنی دانشجوی عادی و استعداد درخشان)) در هر یک از این کلاس ها نمره **منفی** دارد! استفاده از مفاهیم شیء گرایی، تفاوت در Private یا Public بودن و استفاده از Static در موارد مورد نیاز باید کاملا رعایت شود. شرط پیاده سازی getter و setter هم چنان پابرجاست. جاهایی که استفاده از چند ریختی (polymorphism) امکان پذیر است باید با این روش نوشته شود.

اجرا: ادمین از طریق منو(که شما طراحی کرده اید) می تواند یک (یا چند) دانشجو را اضافه کند. وظیفه شما این است که کانستراکتور های دو کلاس ذکر شده(+ کلاس های مضاعف مربوط به دانشجوی کارشناسی) را جوری طراحی کنید که به شکل خودکار بفهمد که باید شیء از نوع دانشجوی استعداد درخشان کارشناسی باشد یا دانشجوی عادی درخشان کارشناسی باشد یا دانشجوی ارشد و به شکل کانستراکتور های اوورلود شده باشند (با توجه به متغیر ها نوع شیء خود به خود تشخیص داده می شود)! تمامی ویژگی هایی که باید وجود داشته باشند مانند :

- نام و نام خانوادگی (مثلا حامد فرجی)
- شماره دانشجویی (یک عدد 4 رقمی یکتا که دو رقم اول مربوط به سال ورود می باشد)
- کد ملی (وقتی ادمین دانشجو اضافه می کند کد ملی همان رمز عبور است)
- رمز عبور (در ابتدا همان کد ملی است اما در ادامه دانشجو می تواند آنرا تغییر دهد. **پس باید در منوی دانشجو این حالت تعبیه گردد**)
- حداکثر تعداد واحد در هر ترم (که عددی **ثابت** و برابر 20 می باشد)
- درس های گذرانده شده (تمامی دروسی که دانشجو گذرانده است که لیستی شامل اشیاء از جنس کلاس درس کارشناسی یا کارشناسی ارشد می باشد و بعد از وارد شدن نمره درس های انتخاب شده توسط ادمین به این لیست اضافه می شود – مخصوص کلاس فرزند)
- درس های انتخاب شده در ترم جاری (تمامی دروسی که برای دانشجو انتخاب شده است که لیستی شامل اشیاء از جنس کلاس درس کارشناسی یا کارشناسی ارشد می باشد – مخصوص کلاس فرزند)
- رشته دانشجو (مثلا مهندسی کامپیوتر که یک شیء از نوع کلاس رشته می باشد)
- رشته تحصیلی دوم دانشجو (مثلا مهندسی کامپیوتر که یک شیء از نوع کلاس رشته می باشد و مخصوص دانشجوی کارشناسی استعداد درخشان می باشد – مخصوص کلاس فرزند سطح ۲)
- معدل (لیستی از معدل ترم های مختلف دانشجو و تعداد واحد آن. می توان از HashMap برای پیاده سازی این مورد استفاده کرد که در آن کلید یک عدد ۳ رقمی است (مثلا 971 که نشان دهنده ترم 1 سال 97 می باشد) و مقدار یک لیست شامل معدل ترم (جایگاه 0 لیست) و واحد گذرانده (جایگاه 1 لیست) می باشد – مخصوص کلاس فرزند) .
- ترم جاری (که یک عدد سه رقمی است مثلا 971 که نشان دهنده ترم 1 سال 97 می باشد. این مورد برای آپدیت کردن معدل ترم جاری در هربار ورود نمره یک درس است در نظر گرفته شده است)

```
Map<Integer, List<float>> GPAs= new HashMap<>();
List<Integer> GPAandUnits = new ArrayList<>();
GPAandUnits.add(18.64);// معدل در ترم
GPAandUnits.add(15);// تعداد واحد گزرانده در ترم
GPAs.put(972, GPAandUnits);// معدل یک ترم خاص
```

پس از اینکه این دانشجو ساخته شد در ابتدا ویژگی هایی مانند درس های گذرانده شده، درس های انتخاب شده در ترم جاری و معدل مقدار ندارند. اینجاست که وظیفه انتخاب واحد اضافه می شود. ادمین باید با رعایت تمام شرط ها(پیش نیازی(با توجه به درس های گذرانده شده) – تداخل(با توجه به درس های اتخاب

شده) – ظرفیت خالی – رشته و سال ورودی و حداکثر تعداد واحد مجز برای دانشجو) بتواند درس برای دانشجو بر دارد.

قسمت دیگر در منو مربوط به وارد کردن نمره است که برای یکی از درس های انتخاب شده توسط متدی در این کلاس نمره وارد می شود و بلافاصله با توجه به ترم دانشجو معدل مربوطه از لیست برداشته شده و با توجه به نمره و واحد درس آپدیت می شود. فرمول:

(تعداد واحد درس + تعداد واحد ترم) / ((تعداد واحد درس * نمره درس) + (تعداد واحد * معدل ترم)) = معدل جدید

تعداد واحد درس + مقدار قبلی تعداد واحد ترم = تعداد واحد ترم

پس از بروز کردن مقادیر بالا درس از لیست دروس ترم جاری حذف و به دروس گذرانده شده اضافه می شود.

در این کلاس ۲ مقدار قابلیت ادیت باید داشته باشند:

- رمز عبور(که توسط خود دانشجو می تواند تغییر کند)
- ترم جاری (**که توسط ادمین و با شرط خالی بودن لیست دروس انتخاب شده در ترم می تواند تغییر کند)**

در بخشی ار منو باید ادمین بتواند تمامی اطلاعات مربوط به دانشجویان را ببیند. و ازطرفی هر دانشجو هم باید بتواند درس های انتخاب شده و معدل ترم های خود را ببیند.

قسمت اضافه شده: اطلاعات تمامی دانشجویان بلافاصله بعد از ساخته شدن شیء باید در فایل Students.txt ذخیره و هنگام نمایش هر گونه اطلاعات مربوط به دانشجو، از همان فایل خوانده شود. در این بخش استفاده از آبجکت های ساخته شده اشتباه است چرا که با اجرای دوباره برنامه این شیء ها قطعا وجود نخواهند داشت. هدف ذخیره سازی اطلاعات گذشته و بازیابی آنها می باشد!

نمره اضافی:

- دریافت اطلاعاتی چون ساخت دانشجو و انتخاب واحد و وارد کرد معدل و تغییر ترم و تغییر رمز و نمایش اطلاعات مخصوص ادمین به او و اطلاعات مخصوص دانشجو به خود دانشجو توسط GUI

3. پیاده سازی: کلاس استاد استخدام شده و استاد موقت باید کاملا جدا از هم بوده و با استفاده از مفاهیم ارث بری پیاده سازی شوند. تکرار ویژگی های پدر(کلاس استاد) در هر یک از این کلاس ها نمره **منفی** دارد! استفاده از مفاهیم شیء گرایی، تفاوت در Private یا Public بودن و استفاده از Static در موارد مورد نیاز باید کاملا رعایت شود. شرط پیاده سازی getter و setter هم چنان پابرجاست. جاهایی که استفاده از چند ریختی (polymorphism) امکان پذیر است باید با این روش نوشته شود.

اجرا: ادمین از طریق منو(که شما طراحی کرده اید) می تواند یک (یا چند) استاد را اضافه کند. وظیفه شما این است که کانستراکتور های دو کلاس ذکر شده را جوری طراحی کنید که به شکل خودکار بفهمد که باید شیء از نوع استاد ایتخدام شده است یا استاد موقت و به شکل کانستراکتور های اوورلود شده باشند (با توجه به متغیر ها نوع شیء خود به خود تشخیص داده می شود)! تمامی ویژگی هایی که باید وجود داشته باشند مانند :

- نام و نام خانوادگی (مثلا علی علوی)
- کد استاد (عدد ۲ رقمی یکتا)
- رمز عبور (هر چیزی)
- تعداد واحد ارائه شده توسط استاد (در ابتدا صفر است و همزمان با وارد شدن شیء استاد به عنوان استاد ارائه دهنده یک درس بروز می شود – مخصوص کلاس فرزند)
- رتبه استادی (رتبه 1 یا 2 یا 3 – مخصوص کلاس فرزند)
- سنوات (مثلا 3 سال برای استاد استخدامی و 64 ساعت برای استاد موقت – مخصوص کلاس فرزند)
- درس های ارائه شده (در ابتدا این لیستی شامل اشیاء از جنس کلاس درس خالی است و بلافاصله با ورود شیء استاد به عنوان ایتاد ارائه دهنده درس، این لیست بروز می شود – مخصوص کلاس فرزند)

حالا پس از اینکه این استاد ها ساخته شدند باید قابلیت دیدن اطلاعات هر استاد برای خودش وجود داشته باشد.

قسمت اضافه شده: اطلاعات تمامی استاد ها بلافاصله بعد از ساخته شدن شیء باید در فایلProfessors.txt ذخیره و هنگام نمایش هر گونه اطلاعات مربوط به استاد، از همان فایل خوانده شود. در این بخش استفاده از آبجکت های ساخته شده اشتباه است چرا که با اجرای دوباره برنامه این شیء ها قطعا وجود نخواهند داشت. هدف ذخیره سازی اطلاعات گذشته و بازیابی آنها می باشد!

نمره اضافی:

- دریافت و نمایش اطلاعات توسط GUI

4. پیاده سازی: کلاس رشته باید با رعایت مفاهیم شیء گرایی ساخته شود.

اجرا: ادمین از طریق منو(که شما طراحی کرده اید) می تواند یک (یا چند) رشته را اضافه کند. تمامی ویژگی هایی که باید وجود داشته باشند مانند :

- نام رشته (مثلا مهندسی کامپیوتر، برق، مکانیک)
- ترم جاری (که یک عدد سه رقمی است مثلا 971 که نشان دهنده ترم 1 سال 97 می باشد)
- معدل سال گذشته (از روی میانگین معدل های دانشجویان در سال گذشته. اگر ترم جاری 972 یا 971 باشد معدل سال 96 مورد نیاز است. این مورد با بروز شدن معدل دانشجویان هم باید بروز شود)
- دانشکده (مثلا فنی)
- دروس اجباری (از لیست دروس با ID اضافه می شود و خود لیستی شامل اشیاء از جنس درس است و همزمان باعث مقدار دهی به ویژگی رشته در کلاس درس می گردد)
- دروس اختیاری (از لیست دروس با ID اضافه می شود و خود لیستی شامل اشیاء از جنس درس است و همزمان باعث مقدار دهی به ویژگی رشته در کلاس درس می گردد)

حالا پس از اینکه این رشته ها ساخته شدند باید قابلیت دیدن اطلاعات رشته ها برای ادمین وجود داشته باشد.

قسمت اضافه شده: اطلاعات تمامی رشته ها بلافاصله بعد از ساخته شدن شیء باید در فایل Majors.txt ذخیره و هنگام نمایش هر گونه اطلاعات مربوط به رشته، از همان فایل خوانده شود. در این بخش استفاده از آبجکت های ساخته شده اشتباه است چرا که با اجرای دوباره برنامه این شیء ها قطعا وجود نخواهند داشت. هدف ذخیره سازی اطلاعات گذشته و بازیابی آنهایت می باشد!

نمره اضافی:

- دریافت و نمایش اطلاعات توسط GUI

منوی پیشنهادی:

1. Admin=>
    - 1.Professor=>
        - 1. Add
        - 2. Show (نمایش اطلاعات تمامی استاد ها)
        - 9.Exit (یک لول بره عقب)
    - 2. Course =>
        - 1.Add
        - 2.Show (نمایش اطلاعات تمامی درس ها)
        - 9.Exit (یک لول بره عقب)
    - 3.Major=>
        - 1.Add
        - 2.Show (نمایش اطلاعات تمامی رشته ها)
        - 3.Add Course => (by ID)
            - 1. Mandatory
            - 2. Voluntary
        - 9.Exit (یک لول بره عقب)
    - 4.Student=>
        - 1.Add
        - 2.Show(نمایش اطلاعات تمامی دانشجو ها)
        - 3.Take Course
        - 4.Add Student's Subject Score
        - 5.Edit Semester
        - 9.Exit (یک لول بره عقب)
    - 9.Exit (یک لول بره عقب)
2. Course => (by ID)
    - 1.Show Related Students
    - 2.Edit Capacity
    - 9.Exit (یک لول بره عقب)
3. Student=>(by ID and Password)
    - 1.Show My Courses And Grades
    - 2.Edit Password
    - 9.Exit (یک لول بره عقب)
4. Professor=>(by ID and Password)
    - 1.Show My Info
    - 9.Exit (یک لول بره عقب)

نکته 1: ترتیب ساخت اشیاء بدین شکل است=> استاد >> درس >> دانشجو و رشته >> دانشجو. این برای جلوگیری از بروز ارور در روند کار هست. زیرا وجود برخی اشیاء مثل استاد در ساخت شیء درس نیاز است.

نکته 2: هر تغییری که بر روی هر شیء از هر کلاس اتفاق می افتد بلافاصله باید در فایل هم اعمال گردد.

نکته 3: با زدن تمامی بخش های اختیاری نمره شما 25 تا 30 درصد افزایش میابد.

نکته 4: برخی از جزئیات گفته شده صرفا برای دادن ایده می باشد و به هر روش دیگری که کد زده شود و نیاز های صورت سوال برطرف گردد (با شرط رعایت تمامی مفاهیم شیء گرایی) نمره کامل به شما داده می شود. سعی کنید اشتباه فاحش نکنید و اکثر ویژگی ها را پیاده سازی کنید تا نمره کامل بگیرید.

نکته 5: بخش های مربوط به محاسبه معدل و منو جداگانه برای دروس اجباری و اختیاری جزء بخش های اختیاری می باشد.

نکته 6: استفاده از کد سمپل قرار داده شده برای تمرین 2 کاملا آزاد است.

نکته 7: اجباری در استفاده فرمت txt در فایل نیست و می توانید از هر فرمتی استفاده کنید.

نکته 8: **استفاده از پایگاه داده کاملا آزاد است.** به شرط رعایت خصوصیات خواسته شده در توضیحات مربوط به ذخیره سازی با فایل.

نکته 9: اضافه کردن کلاس کلاس که متشکل از درس و ساعت و ظرفیت و در انتخاب واحد بجای خود درس، کلاس انتخاب می شود. این بخش نمره اضافه هست.

نکته 10: استفاده از Exeption Handler نمره اضافی دارد.

نکته 11: شرط گرفتن نمره مثبت ها اجرای کامل و دقیق آنها در هر بخش از برنامه می باشد.

موفق باشید