مهندسی اینترنت **تمرین شماره‌ی ۱**

مدرس: **دکتر خامس‌پناه** طراح: علی الهی، محمد مهدی صالحی

مهلت تحویل: دوشنبه ۱۱ اسفند ۱۳۹۹، ساعت ۲۳:۵۵

بلبل‌ستان!

مقدمه

هدف از این فاز، آشنایی با Maven، Git و Unit Testing است که در تمام فازهای بعدی به آنها نیاز خواهید داشت. همچنین، بخشی از منطق اصلی را نیز پیاده‌سازی می‌کنید. توجه داشته باشید که در این فاز نیازی به پیاده‌سازی مفاهیم وب ندارید و در فازهای بعدی به این مفاهیم پرداخته می‌شود.

کلیت پروژه

سیستمی که در طی درس به توسعه‌ی آن خواهید پرداخت یک سیستم انتخاب واحد است. در این برنامه دانشجو لیستی از دروس ارائه شده[1] را مشاهده کرده و درس‌های مورد نیاز خود را به برنامه‌ی هفتگی اضافه می‌کند. دانشجو در طی انتخاب واحد می‌تواند برنامه‌ی هفتگی خود را مشاهده کرده و در نهایت آن را ثبت کند.

گام‌های پیاده‌سازی:

ایجاد پروژه Maven[2]

ابتدا یک پروژه‌ی Maven بسازید و با ساختار ایجاد شده توسط آن آشنا شوید. همچنین، فایل pom.xml و اطلاعات داخل آن را مشاهده‌کنید. پیشنهاد ما برای انجام پروژه‌های این درس، استفاده از Intellij IDEA می‌باشد. برای ایجاد پروژه Maven در محیط توسعه IntelliJ IDEA، می‌توانید از این لینک استفاده‌کنید.

اضافه‌کردن وابستگی[3]های مورد نیاز

[1] Offering

[2] برای اطلاعات بیشتر در مورد Maven می‌توانید به لینک‌های زیر مراجعه کنید.

Maven – Maven in 5 Minutes (apache.org)

Maven – Maven Getting Started Guide (apache.org)

[3] Dependency

داده‌های مورد نیاز برای انجام این فاز، در قالب JSON به شما داده می‌شود. برای خواندن و تجزیه JSON و تبدیل آن به فرمت مورد نیاز از پکیج‌های مخصوص کار با JSON در Java استفاده کنید. در اینترنت جستجو کنید و بهترین پکیج موجود را به وابستگی‌های پروژه خود اضافه‌کنید.

پیاده‌سازی منطق برنامه

در این فاز نیاز است تا تعدادی از عملیات‌های پایه‌ای برنامه خود را پیاده‌سازی کنید. این عملیات در قالب دستوراتی در خط فرمان[4]، به برنامه شما داده خواهند شد. هر دستور، شکل کلی زیر را دارد:

```
command <JSONData>
```

command نشان‌دهنده نام دستور و JSONData داده‌ی مربوط به آن دستور به شکل serialize[5] شده در قالب JSON است. برای استفاده از این داده، باید با استفاده ازکتابخانه‌ای که در فاز قبل به پروژه اضافه‌کردید، آن را deserialize کنید. همچنین، قابل ذکر است‌که JSONData برای همه دستورات وارد نمی‌شود.

ورودی‌ها به صورت JSON به شما داده می‌شوند که در هر بخش نمونه آن‌ها آمده است. پاسخ‌ها نیز باید به صورت JSON در خط فرمان چاپ شوند. فرمت پاسخ‌ها به دو حالت زیر می‌باشد:

```
{
“success” : true,
“data”: <ResponseData>
}
```

```
{
“success”: false,
“error”: <ErrorMessage>
}
```

تمامی موجودیت‌های خود را برای راحتی در حافظه اصلی نگه دارید.
دستوراتی که باید در این فاز پیاده‌سازی کنید، در ادامه آمده‌اند.

[4] CLI (Command Line Interface)
[5] serialization - What is deserialize and serialize in JSON? - Stack Overflow

۱. اضافه کردن درس

اطلاعات درس شامل کد ارائه (code، که ۸ رقمی است)، نام درس (name)، نام استاد (instructor)، تعداد واحدها (units)، زمان برگزاری کلاس (classTime)، زمان برگزاری امتحان (examTime)، ظرفیت (capacity) و لیست پیشنیازها (prerequisites) است. به عنوان مثال، با وارد کردن دستور زیر، درس Internet Engineering با اطلاعات ذکر شده به مجموعه دروس اضافه می‌شود. در نظر داشته باشد که ممکن است چند ارائه از یک کد درس وجود داشته باشد که در زمان برگزاری کلاس یا در استاد درس متفاوت باشند.

نمونه دستور:

```
addOffering {“code”: “81013602”, “name”: “Internet Engineering”, “Instructor”: “Ehsan Khamespanah”, “units”: 3, “classTime”: {“days”: [“saturday”, “Monday”], “time”: “16-17:30”}, “examTime”: {“start”: “2021-9-01T08:00:00”, “end”: “2021-9-01T08:00:00”}, “capacity”: 60, “prerequisites”: [“Advanced Programming”, “Operating Systems”]}
```

۲. اضافه کردن یک دانشجو

این دستور، یک دانشجو را در سیستم ثبت می‌کند. اطلاعات دانشجو عبارتند از شماره دانشجویی (کد ۹ رقمی)، نام و سال ورود.

```
addStudent {“studentId”: “810196123”, “name”: “Ali”, “enteredAt”: “1396”}
```

۳. گرفتن لیست دروس ارائه شده

```
getOfferings {“StudentId”: “810196123”}
```

با وارد کردن این دستور، لیست تمامی دروس ذخیره شده چاپ می‌شود. اطلاعاتی هر درس شامل کد درس، نام درس و نام استاد است. در این لیست باید ارائه‌های یک درس خاص پشت سر هم چاپ شوند. دقت کنید که در این دستور و دستورات بعدی، نیاز است تا شماره دانشجویی در JSONData به برنامه داده شود تا مشخص شود کدام دانشجو در حال اجرای آن دستور است.

نمونه اطلاعات خروجی:

```
“data”: [{“code”: “810125021”, “name”: “Data Structure”, “Instructor”: “Ahmad Ahmadi”},
{“code”: “81013602”, “name”: “Internet Engineering”, “Instructor”: “Ehsan Khamespanah”}]
```

۴. گرفتن اطلاعات یک ارائه

برای خروجی این دستور، درس مورد نظر به همراه تمامی ویژگی‌ها در قالب JSON چاپ می‌شود. اگر درسی با این نام وجود نداشت، صرفاً یک پیام مبنی بر عدم وجود درس به کاربر نمایش داده می‌شود.

نمونه دستور:

```
getOffering  {“StudentId”: “810196123”, “code”: “81013602”}
```

نمونه اطلاعات خروجی:

```
“data”: {“code”: “81013602”, “name”: “Internet Engineering”, “Instructor”: “Ehsan
Khamespanah”, “units”: 3, “classTime”: {“days”: [“saturday”, “Monday”], “time”:
“16-17:30”}, “examTime”: {“start”: “2021-9-01T08:00:00”, “end”: “2021-9-01T08:00:00”},
“capacity”: 60, “prerequisites”: [“Advanced Programming”, “Operating Systems”]}
```

۵. اضافه کردن یک درس به برنامه هفتگی (با داشتن کد درس)

پس از وارد کردن این دستور، درس وارد شده به برنامه هفتگی دانشجو اضافه می‌شود. اگر درسی با این مشخصات وجود نداشت، صرفا یک پیام مبنی بر عدم وجود درس به کاربر نمایش داده می‌شود.

نمونه دستور:

```
addToWeeklySchedule {“StudentId”: “810196123”, “code”: “810125021”}
```

۶. حذف یک درس از برنامه هفتگی (با داشتن کد درس)

با وارد کردن این دستور، درس وارد شده از برنامه هفتگی دانشجو حذف می‌شود. اگر درسی با این مشخصات وجود نداشت، صرفا یک پیام مبنی بر عدم وجود درس به کاربر نمایش داده می‌شود.

نمونه دستور:

```
removeFromWeeklySchedule  {“StudentId”: “810196123”, “code”: “810125021”}
```

۷. مشاهده برنامه هفتگی

```
getWeeklySchedule {“StudentId”: “810196123”}
```

در خروجی این دستور، لیستی از دروس انتخاب شده در قالب JSON نمایش داده می‌شود. دقت کنید که اطلاعات چاپ شده شامل کد درس، نام درس، نام استاد، زمان برگزاری، زمان امتحان و وضعیت درس است. وضعیت درس شامل دو حالت finalized و non-finalized است. اگر دستور finalize (دستور ۸) اجرا شده باشد، درس ها نهایی شده و " status" آن‌ها از non-finalized به finalized تغییر می‌کند.

نمونه اطلاعات خروجی:

```
“data”: {“weeklySchedule”: [{“code”: “81013602”, “name”: “Internet Engineering”, “classTime”: {“days”: [“saturday”, “monday”], “time”: “16-17:30”}, “examTime”: {“start”: “2021-9-01T08:00:00”, “end”: “2021-9-01T11:00:00”, status: “finalized”}, {“code”: “81012501”, “name”: “Data Structure”, “classTime”: {“days”: [“sunday”, “tuesday”], “time”: “9-10:30”}, “examTime”: {“start”: “2021-10-01T08:00:00”, “end”: “2021-10-01T11:00:00”}, status: “non-finalized”]}
```

۸. نهایی کردن برنامه درسی ترم

```
finalize {“StudentId”: “810196123”}
```

با وارد کردن این دستور، برنامه درسی انتخاب شده ثبت می‌شود. ثبت برنامه به معنی "finalized" شدن "status" همه درس های انتخاب شده است اما پیش از ثبت باید بررسی های زیر صورت گیرند:

- مجموع واحد ها باید کمتر از ۲۰ و بیشتر از ۱۲ واحد باشد.
- زمان کلاس ارائه‌ها نباید تداخل داشته باشد.
- زمان امتحان ارائه‌ها نباید تداخل داشته باشد.
- دروس باید ظرفیت کافی داشته باشند.

نمونه خطاها:

نوع خطا	توضیحات
MinimumUnitsError	برای ثبت دروس باید حداقل ۱۲ واحد داشته باشید!
MaximumUnitsError	حداکثر ۲۰ واحد می‌توانید اخذ کنید!
ClassTimeCollisionError <Code#1> <Code#2>	زمان دو ارائه تداخل دارند! ارائه‌ها با کد های code#1 و code#2 با یکدیگر تداخل زمانی دارند.
ExamTimeCollisionError <Code#1> <Code#2>	زمان دو امتحان تداخل دارند! امتحان ارائه‌ها با کد های code#1 و code#2 با یکدیگر تداخل زمانی دارند.
CapacityError <Code>	ظرفیت درس با کد داده شده، تکمیل شده!

برخی دیگر از خطاها که در طی سایر دستورات ممکن است کاربر با آن‌ها مواجه شوند عبارتند از:

نوع خطا	توضیحات
OfferingNotFound	ارائه با این کد وجود ندارد!
StudentNotFound	دانشجویی با این شماره دانشجویی در سامانه ثبت نشده!

آزمون واحد[6]

در این قسمت باید با استفاده از کتابخانه‌ی JUnit برای سناریوهای مختلف نهایی کردن برنامه درسی و مشاهده‌کردن برنامه‌ی هفتگی تست بنویسید. تست‌های شما باید ساختار مناسب Test، Setup و Teardown را رعایت‌کنند.

افزودن پروژه به گیت

ابتدا در سایت DevOps Platform Delivered as a Single Application | GitLab عضو شوید و یک مخزن خصوصی ایجاد کنید. سپس، اکانت ieSpring00 را به پروژه خود اضافه کنید. تمامی تغییرات خود را به گیت اضافه کنید و در نهایت، در مخزن خود بارگذاری کنید. توجه کنید که پروژه شما پس از clone شدن باید به راحتی قابل اجرا باشد. برای تمرین نحوه‌ی کار با گیت توصیه میشود که این لینک و برای آشنایی با شیوه‌ی مناسب کامیت توصیه میشودکه این لینک را مطالعه کنید.

[6] Unit Testing

نکات پایانی

- کافی است که یکی از اعضای گروه Hash مربوط به آخرین کامیت پروژه را در سایت درس آپلود کند. در هنگام تحویل، پروژه روی این کامیت مورد ارزیابی قرار می‌گیرد.
- متاسفانه سایت گیت‌لب برای ایرانیان در دسترس نیست و برای دسترسی به آن می‌توانید از شکن یا FoD استفاده کنید.
- ساختار صحیح و تمیزی کد برنامه، بخشی از نمره‌ی این فاز پروژه‌ی شما خواهد بود. بنابراین در طراحی ساختار برنامه دقت به خرج دهید.
- هدف این تمرین یادگیری شماست. لطفاً تمرین را خودتان انجام دهید. در صورت مشاهده‌ی مشابهت بین کدهای دو گروه، از نمره هر دو گروه مطابق سیاستی که در کلاس گفته شده است کسر خواهد شد.
- سوالات خود را تا حد ممکن در فروم درس مطرح کنید تا سایر دانشجویان نیز از پاسخ آنها بهره‌مند شوند. در صورتی که قصد مطرح کردن سوال خاص‌تری داشتید، از طریق ایمیل با طراحان این فاز پروژه ارتباط برقرار کنید.
- ایمیل طراحان پروژه:

ae.561999@gmail.com
msalehi20@gmail.com