به نام خدا

تمرین سوم درس برنامه‌نویسی پیشرفته

۰. فایل مربوط به توضیحات نحوه ارسال تمرین‌ها را که در مودل قرار دارد، مطالعه کنید.

۱. تمامی فایل‌های کد را به همراه فایل متنی که در قالب pdf است (مورد سوم را بخوانید) به صورت یک فایل آرشیو zip (zip =! rar) که به قالب زیر نام‌گذاری شده باشد، بارگذاری نمایید.

StudentNumber_FirstName_LastName.zip

9031066_Ehsan_Edalat.zip

۲. در سوال‌هایی که ورودی و خروجی مطلوب آن‌ها مشخص شده است، برنامه‌ی شما به صورت ماشینی تصحیح می‌شود. بنابراین رعایت نحوه ورودی‌گرفتن و نمایش خروجی اهمیت بسیاری دارد. دقیقا همان‌طور که از شما خواسته شده است ورودی‌ها را خوانده و خروجی‌ها را تولید کنید.

۳. پاسخ سوالات تشریحی را به صورت تایپ‌شده و در قالب یک فایل pdf (برای کل تمرین) تحویل دهید.

مهلت تحویل: تا جمعه ۳ آبان ۱۳۹۸ ساعت ۲۳:۵۵ شب

سوال اول

جاهای خالی را با کلمه‌ی مناسب پرکنید.

A) An array is a group of _____ (called elements or components) containing values that all have the same _____ .

B) Java class _____ provides the capabilities of array-like data structures that can resize itself dynamically.

C) _____ is used to iterate through a collection and can remove elements from the collection during the iteration.

سوال دوم

مفاهیم زیر را به اختصار توضیح دهید:

Private/Public Access Modifier:

Static:

Final:

Wrapper Class:

Autoboxing and Unboxing:

سوال سوم

در این تمرین می‌خواهیم شبیه‌سازی کلاس‌های یک سیستم مدیریت چت‌روم ساده را پیاده‌سازی کنیم.

- یک کلاس با نام User در نظر می‌گیریم که شامل یک فیلد nickname و یک فیلد email است. همچنین، این کلاس دارای یک تابع createNewMessage(String text) است که به عنوان ورودی یک متن می‌گیرد و خروجی یک پیام از نوع Message را برمی‌گرداند.
- یک کلاس با نام Message در نظر می‌گیریم که شامل یک فیلد text از نوع String به عنوان محتوای پیام، یک فیلد sender از نوع User به عنوان ایجادکننده یا فرستنده پیام و یک فیلد createdDate از نوع Date به عنوان زمان ایجاد پیام است. (کلاس Date در خود زبان جاوا به صورت پیش‌فرض موجود است. برای اطلاعات بیشتر می توانید از این لینک استفاده کنید.)
- یک کلاس با نام Room در نظر می‌گیریم که شامل لیستی از کاربرهای حاضر در آن اتاق و لیستی از پیام‌های فرستاده‌شده در آن اتاق است. این اتاق شامل یک متد برای اضافه‌کردن کاربر جدید و همین‌طور حذف‌کردن کاربر از لیست کاربران است. هنگام اضافه‌کردن کاربر جدید باید بررسی کنید که کاربر تکراری نباشد. تکراری بودن کاربر باید بر اساس ایمیل کاربر بررسی شود. (تکراری‌بودن کاربر را با استفاده از یک متد equals در کلاس User پیاده‌سازی کنید.) علاوه برآن، یک متد برای اضافه‌کردن پیام جدید به اتاق نیز باید پیاده‌سازی شود. ورودی این متد از جنس کلاس Message باید باشد. لازم است که فرستنده پیام در لیست کاربران فعال در آن اتاق وجود داشته باشد. در غیر این صورت نباید پیام اضافه شود. در هنگام حذف‌کردن کاربر از اتاق باید تمام پیام‌های فرستاده‌شده آن کاربر نیز از لیست پیام‌های موجود داخل اتاق حذف شود.
- این متدها را به ترتیب با نام‌های

addNewUser(User userToAdd)

removeUser(User userToRemove)

addNewMessage(Message messageToAdd)

پیاده‌سازی کنید.

- کلاس Room دو متد print نیز دارد که به این صورت باید پیاده‌سازی شوند:
  - متد printUsers که nickname همه کاربران حاضر در اتاق را چاپ می‌کند.
  - متد printMessages به این صورت باید پیاده سازی شود که پیام‌ها به ترتیب زمان ایجادشان باید چاپ شوند یعنی قدیمی‌ترین پیام اول و جدیدترین پیام آخرین پیام باشد. به این صورت که ابتدا تاریخ و سپس نام کاربر فرستنده چاپ می‌شود و بعد از آن پیامی که فرستاده در یک خط چاپ می‌شود. به عنوان مثال :

19 Aug , Hadi = salam

- یک کلاس با نام Messenger نیز وجود دارد که لیستی از اتاق‌ها و لیستی از کاربرها را در خود نگهداری می‌کند و امکان اضافه کردن اتاق جدید و کاربر جدید را دارد. یعنی شامل دو متد addNewUser() و createNewRoom() است. در اضافه‌کردن اتاق‌ها محدودیتی وجود ندارد. ولی در هنگام اضافه‌کردن کاربر جدید به سیستم باید بررسی کنید که کاربر جدید کاربری تکراری نباشد. تکراری بودن کاربران همان‌طور که قبلا گفته شد، با ایمیل بررسی می‌شود. در صورت تکراری بودن کاربر باید پیام تکراری بودن ایمیل نمایش داده شود.
- متدهای getter و setter تمامی کلاس‌ها در صورت نیاز پیاده‌سازی شود.
- برای این تمرین یک کلاس با نام Demo ایجاد کنید و تمام قابلیت‌های برنامه را با ایجادکردن تعداد کافی شئ از هر کدام از کلاس‌ها تست کنید.

سوال چهارم

در این تمرین درس‌های داخل پرتال آموزشی شبیه‌سازی می‌شوند:

- یک کلاس با نام Course وجود دارد که درس‌های داخل پرتال آموزشی را مدل می‌کند. این کلاس دارای فیلدهای نام و برش زمانی اول و برش زمانی دوم است. فیلد نام نماینده اسم درس است. فیلدهای برش زمانی نماینده این است که در در چه بازه‌های زمانی در هفته، درس ارائه می‌شود. همچنین، این کلاس دارای یک متد با نام checkOverlap است. این متد به عنوان ورودی یک Course را می‌گیرد و بررسی می‌کند که این دو درس باهم تداخل زمانی دارند یا خیر. تداخل زمانی دو درس به این صورت تعریف می‌شود: اگر هر کدام از برش‌های زمانی مربوط به کلاس‌ها باهم از نظر روز و یا زمان ارائه درس تداخل داشته باشند، آن دو درس با هم تداخل خواهند داشت.
- یک کلاس با نام TimeSlot وجود دارد که مدل‌کننده برش زمانی/ تاریخی داخل پرتال است. این کلاس دارای فیلدهای روز، زمان شروع و زمان پایان است. نام روزها باید یکی از روزهای هفته (برای مثال Saturday, Sunday, Monday, ...) باشد و زمان‌ها نیز باید اعداد معتبر بین ۰ تا ۲۳ باشد.
- یک کلاس با نام Student وجود دارد که مدل‌کننده دانشجو است. این کلاس دارای فیلد شماره دانشجویی و لیستی از درس‌های اخذشده توسط او است. برای کلاس دانشجو باید یک متد addCourse پیاده‌سازی شود که یک درس به عنوان ورودی بگیرد و به لیست درس‌های دانشجو اضافه کند، توجه کنید که باید بررسی شود که این درس جدید با درس‌های فعلی از نظر زمانی تداخل نداشته باشد. اگر تداخل هست، باید پیام مناسب خطا چاپ شود. علاوه بر آن، یک متد removeCourse برای حذف درس نیز باید پیاده‌سازی شود که درس ورودی را از لیست درس‌های دانشجو حذف می‌کند. لازم به ذکر است که اگر درس ورودی در لیست درس‌های دانشجو وجود ندارد، باید پیام خطای مناسب چاپ شود.
- متد های getter و setter همه کلاس‌ها باید در صورت نیاز پیاده‌سازی شوند.
- برای این تمرین یک کلاس با نام Demo ایجاد کنید و تمام قابلیت‌های برنامه را با ایجادکردن تعداد کافی شئ از هر کدام از کلاس‌ها تست کنید.

سوال پنجم

تمرین شمارش کلمات و کاراکترهای با ویژگی‌های خاص

در این تمرین می‌خواهیم برنامه‌ای مشابه برنامه wordcounter بنویسیم که تعداد کلمات و کاراکترهای یک نوشته ورودی را حساب کرده و چاپ کند. این تمرین یک برنامه تحت CLI است. معمولا به این مدل برنامه‌ها آپشن‌هایی اضافه می‌شود که با استفاده از آنها می‌توان خروجی برنامه و نحوه عملکرد آن را تنظیم کرد، این مدل آپشن‌ها معمولا در دو فرم کوتاه و بلند دسته‌بندی می‌شوند که در فرم کوتاه کاربر با یک خط تیره به همراه یک کاراکتر آن را معلوم می‌کند. مثلا wordcounter –c که نشان می‌دهد برنامه wordcounter به جای عملکرد پیش‌فرض باید تعداد کاراکترها را شمارش کند. همین آپشن در مدل طولانی معادل wordcounter --chars است که دقیقا همان معنی قبل را می‌دهد.

با توجه به این توضیحات برنامه‌ای بنویسید که مشابه این برنامه wordcounter کار کند و آپشن‌های زیر را داشته باشد. این برنامه از طریق ترمینال/کنسول ورودی را می‌خواند و با توجه به آپشن‌هایی که در زیر تعریف می‌کنیم برنامه باید عملکرد صحیح داشته باشد. توجه کنید که شما این آپشن‌ها را در هنگام فراخوانی برنامه خود دریافت می‌کنید. پس برای استفاده از آنها می‌بایست از متغیر args که یک آرایه از Stringهای ورودی به برنامه است در تابع main خود استفاده کنید.

```
public static void main(String[] args) {}
```

در هنگام اجرای برنامه از طریق ترمینال / کنسول هر کلمه‌ای که با فاصله بعد از اسم برنامه وارد کنید، به عنوان یک عضو از آرایه args به تابع main پاس داده می‌شود. مثلا اگر در کنسول دستور java wordcounter --chars را وارد کنید، جاوا برنامه wordcounter را اجرا می‌کند و رشته chars-- را به عنوان عضو اول آرایه args انتخاب می‌کند. مثال دوم اینکه در حالت wordcounter --find a جاوا به ترتیب find-- و a را عضوهای اول و دوم آرایه args در نظر می‌گیرد.

پس از اجرای برنامه با آپشن‌های مشخص شده‌ای که از این args دریافت می‌کنید، برنامه باید آماده دریافت ورودی جدید از طریق Scanner و اجرای برنامه با آپشن‌های مشخص‌شده روی این ورودی جدید scanner باشد.

دو آپشنی که باید برای این برنامه پیاده سازی کنید به شکل زیر هستند:

۱. آپشنی که تعداد کاراکتر ورودی را برگرداند:

اگر c– یا chars-- به عنوان آپشن وارد شود باید برنامه شما تعداد کاراکترهای ورودی scanner را حساب کرده و برگرداند.

اگر بعد از c– یا chars-- به عنوان آپشن ورودی یک کاراکتر نیز وارد شد باید تعداد آن کاراکتر موجود در ورودی scanner را حساب کند و برگرداند . مثلا chars a-- که باید تعداد کاراکتر های a موجود در نوشته را برگرداند.

۲. آپشن بعدی جستجو برای پیداکردن یک رشته در ورودی است که با آپشن find stringToSearch-- مشخص می‌شود که باید در ورودی scanner که بعدا از کاربر می‌گیرید باید رشته stringToSearch را جست‌وجو کند. در صورت پیداشدن، مقدار true و در غیر این صورت مقدار false را چاپ کند. آپشن f– مدل کوتاه برای همین آپشن است و باید پیاده‌سازی شود. توجه کنید

که طبیعتا آپشن find-- و f– بدون ورودی stringToSearch بی‌معنی خواهد بود و برنامه باید عملکرد مناسب را داشته باشد و اعلام خطا کند.

۳. برنامه در حالتی که هیچ آپشنی به آن داده نشده است باید تعداد کاراکترها را شمارش کند. (مشابه زمانی که آپشن chars-- یا c–) به آن داده شده است.

چند مثال از ورودی و خروجی برنامه :

java wordcounter
INPUT -> salam
OUTPUT -> chars=5

java worcdounter –c a
INPUT -> salam
OUTPUT -> num of a : 2

java wordcounter --find ala
INPUT -> salam
OUTPUT -> true

Java wordcounter –f
ERROR

Java wordcounter –-chars –f a
INPUT -> salam
OUTOUT -> chars : 5
true

java wordcounter –c b –f g
INPUT -> abgcb
OUTPUT -> num of b : 2
true