آزمون نرم‌افزار

تمرین شماره‌ی ۱

مدرس: **دکتر خامس‌پناه**

طراحان: علی اخگری، علی کرامتی

مهلت تحویل: ۱ آبان ، ساعت ۲۳:۵۹

مقدمه

هدف از این پروژه کسب آشنایی با Unit Testing و چارچوب JUnit می‌باشد. Unit Testing یک مرحله در آزمون نرم‌افزار می‌باشد که در آن کوچکترین بخش‌های قابل آزمایش یک برنامه، به طور جداگانه، برای عملکرد صحیح مورد بررسی قرار می‌گیرند. همچنین JUnit یک چارچوب آزمون نرم‌افزار در زبان جاوا می‌باشد که به برنامه‌نویسان این امکان را می‌دهد تا با یک روش استاندارد تست‌های خود را بنویسند و آن‌ها را اجرا کنند.

بخش اول ـ پیاده‌سازی Unit Test

برای شروع، یک دایرکتوری به نام "test/java" در دایرکتوری "src" ایجاد کنید. خوب است تست‌هایی که برای کلاس‌های مختلف می‌نویسید را بر اساس نام کلاس در فایل‌های جداگانه پیاده‌سازی کنید. به عنوان مثال، اگر می‌خواهید برای کلاس "Comment" از دایرکتوری "model" تست بنویسید، خوب است یک دایرکتوری به نام "model" در "test/java" ایجاد کنید و یک کلاس به نام "CommentTest" در آن قرار دهید.

در ادامه، آزمون‌های یونیت مربوط به متدهای موجود در کلاس‌های Commodity، Comment و User را پیاده‌سازی کنید. توجه داشته باشید که بعضی از متدها قابلیت تست شدن با روش parameterized را دارند که در این صورت از این روش استفاده کنید. اطمینان حاصل کنید که کارکردهای مختلف این متدها به درستی و حداقل یک‌بار آزموده شده باشند.

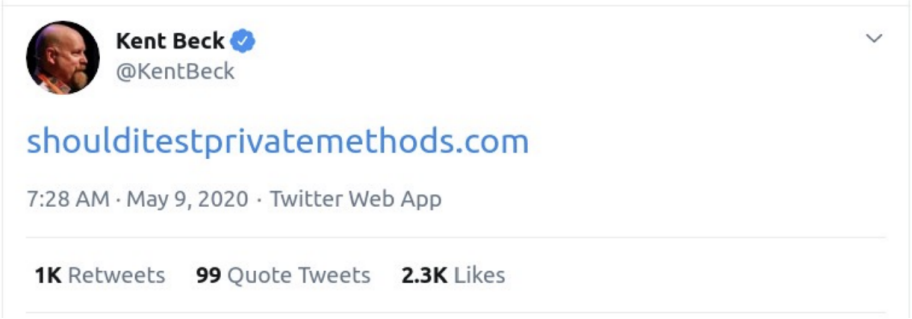
در پیاده‌سازی تست‌ها به موارد زیر توجه فرمایید:

- طبق اسلایدهای درس، **حتما از کتابخانه JUnit 5 استفاده کنید.**
- از نوشتن تست‌های تکراری خودداری فرمایید.
- سعی کنید تست‌ها را مستقل از یکدیگر بنویسید.
- سعی کنید هر تست را فقط برای آزمودن یک بخش از کد پیاده‌سازی کنید.

بخش دوم - گزارش کار

سوال اول

تست کردن متدهای Private بحث برانگیز است. این مساله را تحلیل کنید و فرض کنید به این مسئله برخورد کرده‌اید، بگویید تصمیم شما چیست و چرا؟ کنت بک (از پیشروان TDD) با اشتراک این لینک در توییتر، نظر خودش را در این باره تصریح کرده است. با جستجوی بیشتر می توانید نظرات دیگران را نیز در این زمینه مشاهده نمایید.



سوال دوم

توضیح دهید آیا می‌توان با استفاده از یونیت تست، از درستی یک کد Multi-threaded اطمینان حاصل کرد؟

سوال سوم

مشکلات هر یک از آزمون‌های زیر را در صورت وجود بیان کنید و راه‌حل(های) احتمالی برای تصحیح هر یک از آن‌ها را ارائه دهید.

```
@Test
public void testA() {
    Integer result = new SomeClass().aMethod();
    System.out.println("Expected result is 10. Actual result is " + result);
}
```

```
@Test
public void testC() expects Exception {
    int badInput = 0;
    new AnotherClass().process(badInput);
}
```

```
@Test
public void testInitialization() {
    // Initialize the configuration and resources
    Configuration.initialize();
    ResourceManager.initialize();
    // Perform assertions to validate the initialization
    // ...
}

@Test
public void testResourceAvailability() {
    // Check if a specific resource is available
    boolean isResourceAvailable = ResourceManager.isResourceAvailable("exampleResource");
    assertTrue(isResourceAvailable);
}
```

نکات پایانی

- پروژه در قالب گروه‌های **حداکثر دو نفره** انجام شود.
- برای پیاده‌سازی، ابتدا پروژه را از این لینک **clone** کرده و سپس یک repository در صفحه شخصی خود ایجاد کرده و تغییرات لازم را روی آن اعمال کنید.
- گزارش کار در قالب یک pdf در صفحه درس آپلود شود. توجه داشته باشید که لازم **نیست** کد آزمون‌های پیاده‌سازی شده را در گزارش بیاورید.
- برای تحویل کافیست یکی از اعضای گروه گزارش پروژه که ابتدای آن شامل آخرین شناسه کامیت نیز می‌باشد را در صفحه درس بارگزاری نماید.
- لطفا کاربر **SoftTest-ut** را به repository خود اضافه کنید.
- هدف این تمرین یادگیری شماست. لطفا 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.