1 Unit Testing(15 marks)

1. Imagine that you plan to build a Ed-tech platform based out of UIU where different people can come together and offer to sell their particular skill by the means of selling courses. Other Users can signup into the system and subscribe to courses and watch the content.Now, consider the following class diagrams for a simplified version of the system. The three diagrams below represent the user, course and video contents of a course. Your job is to write unit test cases for the methods of this classes according to the given specifications.

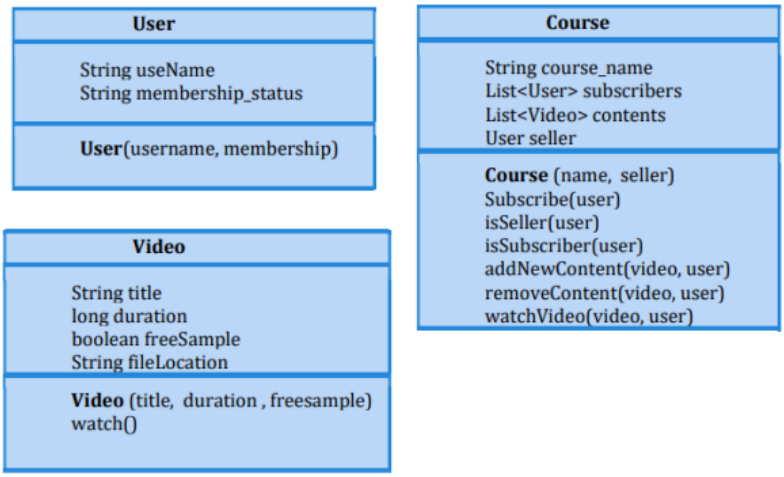

Figure 1: Class diagrams for Ques-1

(a) First, you need to test the *watchVideo* method from Course class. Each user has a membership status which is a String that can be either 'Free' or 'Premium'. If a user is a premium member he should be able to watch any video if he is subscribed to the course. if the user is a free member he can enjoy only the free sample videos. So, you need to write three unit tests to check the following three scenarios. $[4 + 4 + 4 = 12]$

- Test Case 1: Check if the *watchVideo* works properly when it is called by a 'premium' user who is subscribed to the course.Set up a course and add two videos(one is a free sample another one is not) to it. The user under test should be able to watch both videos(i.e. watchvideo() returns *true*).
- Test Case 2: Test the same method for a user with a 'Free' user. Again set up a course with two videos like test case 1. This time however, the user is able to only watch the freesample video.
- Test Case 3: Now run test case 2 without subscribing to the course. The system should throw *NotaSubscriber.class* error with message *"You cannot add a watch a video without subscribing first"* .

Devise executable test cases for all of the above specifications for this method in the *JUnit* notation.

(b) Now you need to conduct a performance test on the same *watchVideo* method which returns true if the video is able to successfully and returns false otherwise. Write an executable test case in JUnit notation that tests whether the method is able to load the video within a 3 second time limit. [3]

```java
@Test
public void testWatchVideoPremiumUserSubscribed() {
    // Create a premium user
    User premiumUser = new User("PremiumUser", "Premium");

    // Create a course
    Course course = new Course("CourseName", premiumUser);

    // Create two videos, one with free sample and one without
    Video freeSampleVideo = new Video("Free Sample Video", 10, true);
    Video premiumVideo = new Video("Premium Video", 20, false);

    // Add videos to the course
    course.addNewContent(freeSampleVideo, premiumUser);
    course.addNewContent(premiumVideo, premiumUser);

    // Premium user should be able to watch both videos
    assertTrue(course.watchVideo(freeSampleVideo, premiumUser));
    assertTrue(course.watchVideo(premiumVideo, premiumUser));
}
```

```java
@Test
public void testWatchVideoFreeUserSubscribed() {
    // Create a free user
    User freeUser = new User("FreeUser", "Free");

    // Create a course
    Course course = new Course("CourseName", freeUser);

    // Create two videos, one with free sample and one without
    Video freeSampleVideo = new Video("Free Sample Video", 10, true);
    Video premiumVideo = new Video("Premium Video", 20, false);

    // Add videos to the course
    course.addNewContent(freeSampleVideo, freeUser);
    course.addNewContent(premiumVideo, freeUser);

    // Free user should only be able to watch the free sample video
    assertTrue(course.watchVideo(freeSampleVideo, freeUser));
    assertFalse(course.watchVideo(premiumVideo, freeUser));
}
```

```java
@Test
public void testWatchVideoFreeUserNotSubscribed() {
    // Create a free user
    User freeUser = new User("FreeUser", "Free");

    // Create a course
    Course course = new Course("CourseName", freeUser);

    // Create a premium video (no free sample available)
    Video premiumVideo = new Video("Premium Video", 20, false);

    // Add the premium video to the course
    course.addNewContent(premiumVideo, freeUser);

    // Attempt to watch the premium video without subscribing (expecting an exception)
    Exception exception = assertThrows(NotaSubscriber.class, () -> {
        course.watchVideo(premiumVideo, freeUser);
    });

    // Check the error message
    assertEquals("You cannot watch a video without subscribing first", exception.getMessage());
}
```

```java
@Test(timeout = 3000) // Set a timeout of 3000 milliseconds (3 seconds)
public void testWatchVideoPerformance() {
    // Create a user (free or premium, depending on the scenario you want to test)
    User user = new User("TestUser", "Premium");

    // Create a course
    Course course = new Course("CourseName", user);

    // Create a video with a simulated long loading time (e.g., 4 seconds)
    Video slowVideo = new Video("Slow Video", 4000, false);

    // Add the slow video to the course
    course.addNewContent(slowVideo, user);

    // Attempt to watch the slow video
    boolean result = course.watchVideo(slowVideo, user);

    // Ensure that the result is false (video loading took longer than 3 seconds)
    assertFalse(result);
}
```