BLG 475E - Software Quality and Testing
2023-24 Spring Term
Homework #1

Due: March, 29 23:59

Introduction

In this assignment, you will learn how to implement and create unit test cases using JUnit framework.

For any issues regarding the assignment, please contact Sultan Çoğay (**cogay@itu.edu.tr**).

Implementation Notes

The implementation details below are included in the grading:

1. Please follow a consistent coding style (indentation, variable names, etc.) with comments.
2. Please follow unit testing best practices summarized in the lecture notes. e.g. Test method names must be meaningful, and they should provide a clear idea what the test is for.

Submission Notes

1. You should compress all your necessary files into an archive file (`.zip`). Please write your name and ID on the top of each document that you will upload as in following format:
```
/* @Author
* Student Name:  <student_name>
* Student ID: <student_id>
*/
```
2. Submissions are made through Ninova system **only**, and have a strict deadline. Assignments submitted after the deadline will not be accepted.
3. This is not a group assignment and getting involved in any kind of cheating is subject to disciplinary actions. Your homework should not include any copy-paste material.
4. You can not publish your homework (on Github, soical media, personal blogs etc.) before it has graded.

1 Questions

1.1 Warm-up: E-mail Address (30 pts.)

An email address is a unique identifier for an email account. It allows messages to be sent and received over the internet. Typically structured as local-part@domain, the email address consists of two main parts:

- Local-part: This is the part before the @ symbol. It identifies a specific mailbox within a domain. It can contain letters, numbers, and certain special characters like periods, underscores, and hyphens.
- Domain: This part follows the @ symbol and specifies the mail server that hosts the email account. Usually represents the email provider's website (e.g., gmail.com, outlook.com) or a specific organization's domain in the case of corporate email addresses.

You are given a source file called EmailValidator.java that checks whether an e-mail address is valid or not according to the given rules, and a test file called EmailValidatorTest.java for the corresponding source file. **Analyze** EmailValidatorTest.java for any bad practices or unusual convention examples, **correct** them, and **briefly state** (adding one-line comments) why it is fixed.

Evaluation criteria: Your test file will be evaluated based on the manual review on unit tests' quality.

1.2 Unit testing: Book Rating (70 pts.)

You are given a source file called BookRating.java that includes methods for saving and rating your read books. Investigate the given Java class and **implement** test cases BookRatingTest.java for the given methods of this class. Details and definitions of the methods are given in the code.

Investigate the given Java class and **implement** test cases for the given methods of this class considering both black box (equivalence class partitioning) and white box testing strategies (coverage) as well as good testing practices summarized in the lecture notes.

Evaluation criteria: Your test file will be evaluated based on the achieved branch coverage and manual assessment of unit tests' quality and effectiveness in detecting bugs. Test files with less than **70%** branch coverage will get only partial points.

Info: The source code given in this assignment is implemented using Intellij IDEA (as an IDE) and JUnit5 (as a testing framework). In order to observe branch coverage in Intellij IDEA, don't forget to activate tracing option and run your tests *with coverage*. See documentation for more information.