SOFTWARE ENGINEERING

(WEEK 09)

UNIT TESTING

In this week's lesson you will practice writing automated test programs called unit tests that test the simplest units of a program, such as methods. Choosing a language or tool is not as important as understanding what it means and how to create test cases. The C# language and MS Test framework were chosen to illustrate this lesson, however you can choose any other language or framework as you like, for example using Java and JUnit 5.

In addition to submitting your assignment on the Google Classroom system, the working content of today's assignment needs to be committed to your github repository. You need to make sure your github repository is available until at least the end of this semester.

Exercise 1. Write unit tests to test the functionalities of the **Student** and **StudentService** classes. You must write enough test cases based on the description above each method. Make sure the test coverage reaches 100%.

- After completing each unit test method, you need to create a commit to push the source code to the github repository on the main branch.

Exercise 2. Based on the source code you provided, what functional requirements are currently not satisfied? Write your answer in MS Word or Notepad (e.g. ex2.docx).

- Create a new commit to put the ex2.docx file on the main branch of the github repository.

Exercise 3. Modify the provided source code so that all requirements are satisfied (all test cases are passed).

- Create a new branch named fixed, then make a commit to push the fixed source code to this branch.

INSTRUCTIONS

In the given source code, you will find two files Student.java and StudentService.java containing noted methods similar to the example below.

```
/**
 Trả về xếp loại của sinh viên theo ký tự A,B,C,D,E
        + [8, 10]: Loại A
        + [7, 8): Loại B
        + [5, 7): Loại C
        + [3.5, 5): Loại D
        + Dưới 3.5: Loại E

    CẦN TỐI THIỂU: 4 testcases
 */
2 usages
public char getLetterScore() {
    if (score >= 8.0) return 'A';
    if (score > 7.0) return 'B';
    if (score > 5.0) return 'C';
    if (score >= 3.5) return 'D';
    return 'E';
}
```

Example of a method under test and its expected outcome in the description

In the example above:

- The comment section is a description of exactly what the method needs to be done.
- The code below is actual based on the description, but may not be 100% as described above.

Your tasks is:

- Based on the description in the comment to write test cases to test the methods.
- Each method may need multiple test cases to test.