**Individual Project: *Grades Calculator***

**Background:**

For this new project, you must develop a simple tool called *GradesCalculator*. Given a database with various data about the students in a class, *GradesCalculator* allows the instructors to get various information about the students and compute their final grade. You will develop the *GradesCalculator* application by following a test-driven development approach such as the one we saw in P4L4, in which development happens in iterations and according to the following cycle: (1) select story cards to implement, (2) define task cards for the selected story cards, (3) write test cases for the classes in the task cards, (4) write code that makes the test cases pass. Specifically, this project will consist of three or four deliverables. (Being in an agile world, we will make some decisions on the fly based on how the project is progressing.)

**Goals:**

* Develop a Java application for assessing students information and compute students’ final grades.
* Get experience with an agile, test-driven process.

**Deliverables:**

**DELIVERABLE 1**

* **provided:**
  + Story card(s)
  + Task card(s)
  + JUnit test cases
* **expected:**
  + Code that makes the test cases pass

**DELIVERABLE 2**

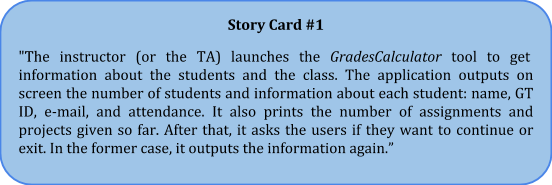
* **provided:**
  + Story cards
  + Task cards
  + JUnit test cases
* **expected:**
  + Code that makes the test cases pass
  + Task cards
  + JUnit test cases

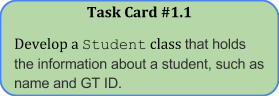
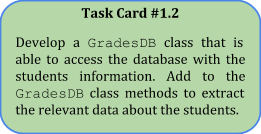
**DELIVERABLE 3**

* **provided:**
  + TBD
* **expected:**
  + TBD

**Deliverable 1: Details**

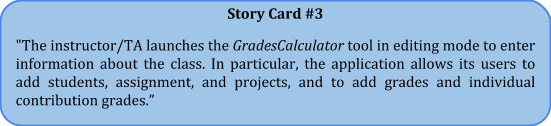
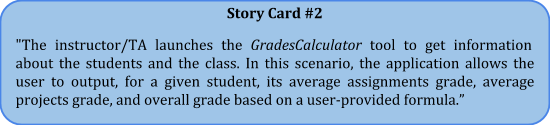
Note: For this deliverable, we will assume to be in the first iteration of the process and that somebody has already done the steps from (1) through (3) described in the background section. You will be provided with a story card, two task cards derived from it, and a set of corresponding test cases, and your job will be to perform step (4): writing code that makes the test cases pass.

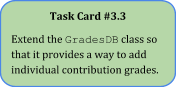
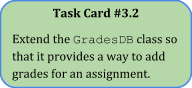
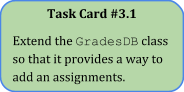
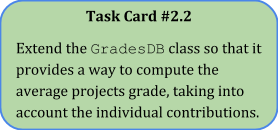
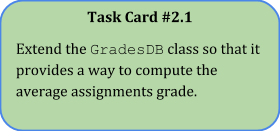


**Deliverable 2: Details**

Your customer decided to assign the development of the user interface for the *GradesCalculator* to Rufus, a professional UI developer. This means that you won’t have to worry about developing that part of Story Card #1. It also means that you will now have to make sure that your code integrates nicely with Rufus’s. In addition, the customer provided you with two additional story cards, and identified, with the help of Rufus, an initial set of corresponding task cards and related test cases. You will have to write code that makes these test cases pass, which means that it should integrate nicely with Rufus’s code. You will also have to write some additional task cards and test cases (but not the application code that makes these latter tests pass). See the instructions below for details.





**Instructions:**

* + An updated GradesDBTest class that contains the test cases created by Rufus. Just like you did for Deliverable 1, you will have to write code that makes them pass, as described below.
  + A golden version of the database that is used by the new tests (GradesDatabase6300-goldenversion.xlsx).
* Keeping in mind the story cards and their corresponding task cards, write code that makes the newly provided test cases pass. Doing so may require you to refactor your existing code, which is expected, as the tests assume that the system behaves in a given way. *Note*: When writing and (possibly) refactoring this code, you don’t have to worry about the parts of the story cards for which there are no corresponding task cards.
* Write task cards to realize the parts of Story Card #3 that refer to the ability of (1) adding students (2) adding projects, and (3) adding grades for a project. Create a directory Cards under directory GradesCalculator and save there your cards, in MD format, with names taskCard3.4.md, taskCard3.5.md, and so on.
* Add to class GradesDBTest test cases for the task cards that you just produced (but do not write the code that makes them pass). As usual, make sure that all test cases have an oracle. *Hint*: you can use the test cases provided by Rufus as an example and a starting point.
* Commit and push your changes, which should consist of the Cards directory and its contents, your code under directory src, and the class GradesDBTest with your additional test cases.
* Paste the commit ID for your submission on T-Square.

**Notes:**

* **Important**: also for this deliverable, you cannot modify the provided test cases; that is, you can add to class GradesDBTest new test methods, **but you cannot modify the existing ones (i.e., you cannot modify anything above the “NO CHANGES ABOVE THIS LINE” comment in the class). Nor you can modify file GradesDatabase6300-goldenversion.xlsx, the golden version of the database.** **Note that, however, the test cases will modify the “regular” database file, GradesDatabase6300.xlsx, as the new functionality of the GradesCalculator involves database updates.**
* Just to clarify, you will not be provided with Rufus’s code (at least for now), so you should not worry about it**. All you have to do is make sure that the code you write makes the provided test cases pass, as they encode the expectations of Rufus’s user interface code.**
* **The average projects grade for a student S in team T should be computed by first multiplying, for each project P, team T’s project grade by the individual contribution of S in P, and then averaging the resulting grades.**
* **When converting real numbers to integers, you should suitably round them up or down based on their decimal part (<.5 round down, >= .5 round up).**