Project #3 – 100 points

Due Date

Monday, November 7, 11:59pm.

Submission

(1) Submit **a single zipped file** to Canvas. The zipped file MUST include the following grading items.
   (a) Source folder **src,** which contains the Java packages you developed. [80 points]
      - The source folder must include all Java files from Project #2, EXCEPT GymManager.java.
      - 3 JavaFX related files, which replace the GymManager.java in Project #2
         - The Java file that contains the `public static void main(),` which has only one method `launch();` you should use the file name **GymManagerMain.java**, or **-2 points.**
         - The controller Java file; ONE controller file ONLY; you should use the file name **GymManagerController.java**, or **-2 points.**
         - The view file; this is the fxml file; ONE fxml file ONLY; you should use the file name **GymManagerView.fxml**, or **-2 points.**
      - MUST use the **@author** tag in the comment block on top of all Java classes, or you will **lose 2 points**.
   (b) Test Design Document. [15 points]
   (c) Javadoc folder. [5 points]

(2) The submission button on Canvas will disappear after **November 7, 11:59pm**. It is your responsibility to ensure your Internet connection is good for the submission. **You get 0 points** if you do not have a submission on Canvas. Submitting the project through the email will not be accepted.

Project Description

You will revamp the user interface for the software you developed in Project 2, to develop a GUI (graphical user interface) with JavaFX. The GUI shall replace the GymManager class in Project 2 and provides the same functionalities developed in Project 2. Project 3 must meet ALL the functional requirements stated in Project 2.

Project Requirement

1. You MUST follow the Coding Standard and Ground Rules posted on Canvas under Week #1 in the "Modules". **You will lose points** if you are not following the rules.
2. You are responsible for following the Academic Integrity Policy. See the **Additional Note #13** in the syllabus.
3. You must preserve all the functionalities developed in Project 2 to pass the test cases.
4. Each Java class must go in a separate file. **-2 points** if you put more than one Java class into a file.
5. You MUST include all the Java classes from Project 2 and use them in this project, EXCEPT the GymManager class and RunProject2 class. You will **lose 5 points** for each class not used. You will **lose 10 points** if you use GymManager.java. **NOTE**, if you lose points in Project 2, you must fix the problems, or you will lose points again for the same cause!
6. You MUST follow all the project requirements listed in Project 2.
7. This project uses the Model-View-Controller (MVC) design pattern. You must use only ONE JavaFX fxml file for the "view", ONE Controller class for the "control", and ALL the classes from Project 2 for the "model". In addition, there will be ONE Java file contains the main() method to "launch" the GUI. **You will get 0 points** if you don't follow the MVC pattern.

8. You can design your own GUI; however, your GUI must include the following JavaFX components.
   - Use at least 4 different Layout Panes, such as BorderPane, GridPane, VBox, Hbox, etc., or **-5 points.**
   - Use a TextArea to display messages or output data, or you will **lose 5 points**. All output MUST be "appended" to the TextArea. That is, if the output is more than the visible part of the TextArea, the user can scroll up and down to see the history of output.
   - Use RadioButton group for single-select items, or **-2 points**, for example, membership types.
9. You MUST set the title of the primaryStage (title for the window.) or **-2 points.**
10. You are **NOT ALLOWED to use System.out** (write to console) or **System.in** (read from console) ANYWHERE in ALL CLASSES, **or you will lose 3 points for each violation, with a maximum of losing 10 points**. This means you MUST modify all the print() methods in the MemberDatabase class to return a string, or define a **toString() method** in the MemberDatabase class. All read and write must be done through the GUI.
11. File input performed for loading members and loading fitness classes using the Scanner class should be done in the MemberDatabase class and ClassSchedule class, respectively, **-3 points** for each violation.
12. You are required to **generate the Javadoc** after you properly commented your code. Your Javadoc must include the documentations for the constructors, private methods and public methods of all Java classes (*.java files.) You **must comment** the **GymManagerMain.java** and **GymManagerController.java** and include them in the Javadoc. DO NOT include the *.fxml file, which is NOT a java file. Generate the Javadoc in a single folder and include it in your project folder to be submitted to Canvas. You are responsible to **double check** your Javadoc after you generated them. The grader will navigate the Javadoc with the "index.html". You will **lose 5 points** for not including the Javadoc, OR, the grader cannot navigate your Javadoc through the "index.html".

System Testing

1. You MUST create a test document and design the test cases for testing Project 3. The test document **is worth 15 points.** Use the test cases provided in Project 2 (Project2_testCases.txt) as a reference to design your test cases.
2. You MUST use the table in the Coding Standard and Ground Rules to organize the test cases, or you **will get 0 points**.
3. Use your test cases to manually test your GUI. All invalid data should be rejected by the GUI. Proper error messages must be displayed in the TextArea. **You will lose 2 points** for each invalid condition not rejected, or error message not properly displayed in the TextArea.
4. You are responsible to thoroughly test your software with the test cases you designed. Your software must always run in a sane state and **should not crash in any circumstances**. The grader, as a user of your software, will try to produce exceptions while running your GUI. You must handle all exceptions. Your software shall continue to run until the grader stops the execution or closes the GUI window. **You will lose 2 points** for each exception not caught.