ProgSD Lab Exam (October 2021) general feedback

- The lab exam was an open book.
- In general, most of you did well because the codes were marked manually. Note that if we had used automatic marking, many would have failed.
- The average for Python is 18.36 out of 25
- The average for Java is 17.77 out of 25
- The mark distribution for the Lab Exam (Python and Java) is 50 to 2
- The grade distribution for the overall ProgSD course is A1 to D3.
- We ran your code and looked if the output was correct, then you would get the mark.

Python: We had already covered all these topics. Many students lost marks due to submitting incomplete or partial solutions or the wrong file format. Students lost 0.5 or 1 mark because the code was not commented well enough.

Task 1a:
It was about creating a program that displayed the screen provided. When run, if the windows appeared as in the example provided, you would get **4 marks**.

The markers would enter the data in the program window as per field requirements. The marker would then click on 'Save to DB', then 'Save to file' so that the **SAME data** is saved to a DB and a file at the same time without filling the form again. As long as it ran without a problem and you called the DB 'movies' as instructed, you got full **8 marks**.

The marker would check that data was really saved to DB. Regardless of how many times "Display movies" is clicked, movies are displayed with all the details as instructed and presented as in the screenshot example, and only one copy of each movie is displayed. This shows that the 'display movies' button has been configured correctly, and it is not duplicating the output of the database in the display box (**3 marks**)

The marker would also click on 'Display Directors' to see that the same movies (only title and director names) are displayed **3 marks**. If one of the fields is missing or not displayed, as shown in the screenshot example, you get 1 mark only. If nothing is displayed = 0. (Note: there was no duplication constraint here).

The marker would then use the 'Clear' button to clear both the form and the display boxes **(2 marks).** If it does not = 0; if it only removes from the form = 1 mark; or if it only removes from the display boxes = 1.

Task 1b:
The marker would attempt to save the same movie in the database to verify the duplication constraint, including the message. If a message appeared and when the display button was clicked once there was no duplication, you would get all **4 marks**. This shows that the 'Save to DB' button (function) was configured correctly.

Java: We used automated tests to check your code for all tasks. If a test did not pass, we manually inspected the reason for the failure, in order to assign marks correctly.

Task 1:
Most students got full marks on this task. Marks were deducted if the enumeration was not correctly declared as public or did not include the correct values. One mark was also deducted here if the submitted code was not in the correct package.
Average 2.6/3

Task 2:
Many students got full marks here, and most got a high mark. The main issues were incorrect access modifiers on the fields (they should all be **private**), not correctly throwing an exception on a non-positive value for credits, incorrectly overriding **equals/hashCode**, or including a **toString()** method that was automatically generated by Eclipse/IntelliJ.
Average 6.4/8

Task 3:
Most students got the marks for the basic structure of the **StudentRecord** class, but many students lost marks for not correctly storing, retrieving, and updating grades. The most straightforward implementation used a **Map<Course, Grade>** -- implementations using a set of arrays or lists often did not correctly implement all of the required behaviour.
Average 5.7/8

Task 4:
Many students did not attempt **computeGPA** at all, or only included the method header. For those students who did implement it, many students had bugs in the implementation, especially with an empty list of courses (which should return 0).
Average 2.7/5

Style:
To get the full mark for style, students needed to include at least comments on every source file as well as “interesting” methods. Many students did not include any comments at all and got 0.5/1 on this category.
Average 0.7/1