Lab Report (Exceptions)

I/we the undersigned, promise that the submitted lab report is/are my/our own work. While I/we was/were

free to discuss ideas with others, the work contained is my/our own. I/we recognize that should this not be the case; I/we will be subject to penalties as outlined in the course syllabus.

(By typing in your name below, you agree to Academic Integrity and honesty)

~~Group Name:~~

Name: Nero Hamidi
Red-Id: 827723033

~~Name:~~
~~Red-Id:~~

Reflection:

Write a few lines explaining your lab learnings and answer the following questions

In this lab I got more experience with exceptions within Java. I was able to learn how to throw new exceptions for specific cases within my program that aren't built Java exceptions. Using if-statements and "throw new" I was able to throw exceptions with messages that made sense for the specific error. For example, when the program tried to add a student to class that was full, I was able to detect this using an if statement and throw ClassFullException (a class which was constructed through extending the base Exception class) with a custom message.

1) why a checked exception ClassFullException is used for failed condition - adding a student when the class size is full

When we create a ClassFullException we are able to detect when the class size is full and output an appropriate message (i.e. "Class is full") instead of a generic error message

2) why a checked exception StudentAlreadyEnrolledException is used for the failed condition - the student is already enrolled in the class

When we use StudentAlreadyEnrolledException we can check for a specific failed condition (student already enrolled) instead of a general case, we are able to display a more appropriate error message.

3) what are some other checked exceptions could be thrown from this code (Think at least 2 more)

The UnsupportedOpperationException (UOE) (custom message) and NullPointerException (no custom message) are also thrown in certain conditions. UOE checks to see if the grade level of a student makes sense (non-negative) when trying to add a student to it and output an appropriate error message. NullPointerException is used to check whether the student exists (student entered is not null) before trying any other operation

4) why the unchecked exception (NullPointerException) used for the bad conditions of the student arguments.

The NullPointerException is used as the preliminary checker before any other operations are done. You can't check the attributes of the student or try to add them to a class unless you know they exist (are not null); this is what NullPointerException does.