Assignment 2

Start Assignment

Due Sunday by 11:59pm **Points** 100 **Submitting** a text entry box or a file upload
Available after Sep 11 at 7am

You have been given a template for the UnitsConvertor class:

- **UnitsConvertor.java (https://psu.instructure.com/courses/2259325/files/151516554/download?wrap=1) (https://psu.instructure.com/courses/2259325/files/151516554/download?download_frd=1)**
- **(https://psu.instructure.com/courses/2259325/files/151516554/download?wrap=1)** For those who are not able to open the above file, I also have a ".txt" version of it, here: **UnitsConvertor.txt (https://psu.instructure.com/courses/2259325/files/151516550/download?wrap=1) (https://psu.instructure.com/courses/2259325/files/151516550/download?download_frd=1)** ---> *it is exactly the same program*.

(https://psu.instructure.com/courses/2259325/files/151516554/download?wrap=1)

This program converts imperial to metric and vice versa. For example, if the input is “9 mile”; then the equivalent value will be displayed on the console in metric in mm, cm, m and km. The reverse functionality is also supported by this program. So, if the input is “9 km”, then the equivalent value will be displayed in miles, inches, feet, yard.

Your task for this assignment is:

- Run the UnitsConvertor program. Analyze its internal structure to understand how it functions.
- Create UnitsConvertorTest.java
- Write a minimum of 15 test cases (JUnit 5)
- Attain 95+% source coverage (Coverage As...) to test the different methods inside the class.
  - Your test cases should cover success and failure scenarios. You need to test the methods with different types of inputs. For failure tests, embed the word "invalid" in the method name, e.g., "test**Invalid**InputNumber()"
  - Comment all the test cases.
- *For this assignment you need to submit one file*: UnitsConvertorTest.java
- Do not use additional libraries (e.g., Mockito, PowerMock, etc. Use techniques shown in TestPrimeOrPerfect.java.
  - Use the normalizeExpectedOutput() method to ensure portable test results between Windows, MacOS, and Linux.

Assignment 2 Rubric

Criteria	Ratings		Pts
Source Formatting The code is Consistently formatted.	**5 pts** **Full Marks**	**0 pts** **No Marks**	5 pts
Documentation Documented each test case	**20 pts** **Full Marks**	**0 pts** **No Marks**	20 pts
15+ different cases.	**20 pts** **Full Marks**	**0 pts** **No Marks**	20 pts
95+% coverage from Coverage As => JUnit Test.	**15 pts** **Full Marks**	**0 pts** **No Marks**	15 pts
Did you test both: success and failure scenarios? I expect both types to be covered in your test cases.	**20 pts** **Full Marks**	**0 pts** **No Marks**	20 pts
Did you test with different types of inputs and assertions statements? Minimum 3 different types of assertion statements to be used. I highly encourage you to be as diverse as possible.	**20 pts** **Full Marks**	**0 pts** **No Marks**	20 pts
			Total Points: 100