**Category One – Software Design and Engineering**

**Selected Artifact:**

For this assignment, I selected the artifact I completed in CS 320: Software Testing, Automation, and Quality Assurance to exemplify my ability to migrate a code base using Java to a code base built with Python. The original artifact is constructed of three Java object classes and three Java classes that modify those objects within a simple data structure. Additionally, numerous Junit tests ensure code quality and coverage.

**Justification:**

This artifact was selected for this category due to its emphasis on code quality and coverage. The tests that ensured the cohesiveness of the code seemed like a good method to display proficiency in software design and engineering. By porting the same functionality to a Python-based environment, I would demonstrate how my engineering experience has taught me to adapt to various coding environments. When tasked with implementing a design, I can utilize a variety of tools and select the most effective ones to deliver an effective product.

**Outcome-Coverage Plan:**

With the completion of category one, I can say with certainty that I achieved the course outcomes attributed to this artifact. While converting Java classes into Python, I was able to effectively employ strategies for building environments that enable diverse audiences by providing two separate approaches to a goal and providing benefits and drawbacks for each. I ensured that each method had a corresponding comment to display how I can deliver professional-quality communication thoroughly for my designs.

Through extensive testing and exceptions in this project, I demonstrated my ability to develop a security mindset that anticipates adversarial exploits (such as injections). Converting the code base from one language to another also partly exemplified an ability to use well-founded and innovative tools (using both JUnit and Pytest).

**Reflection:**

I greatly underestimated the amount of effort required to deliver the proposal I had for this module effectively. Some of my greater challenges stemmed from my less familiarity with Python, as I am more familiar with Java. It took me a while to set up an environment on my machine that ensured my Python implementations could build, and I also found it challenging to convert the Java code into Python. Python is a very adaptable language, and trying to ensure it can perform in a structured way, like the Java implementation I had, was difficult. The most prominent example of this is how it took me a good portion of time to determine how to take a Java object and define it using Python. This language doesn’t necessarily need to utilize predefined objects. Once that structure was in place, the final challenge was determining how to ensure a @BeforeAll annotation in Junit can be replicated using Pytests. The solution I utilized is described in more detail with the comments I left in the code. Mainly, I learned that my port from Java to Python doesn’t necessarily have to be one-to-one in its design and implementation. It may be more beneficial to play to both languages’ strengths.