# June Mwenda

## Question 1

### Part A

High cohesion. This would mean all its methods and attributes work together to perform a single and clearly defined purpose helping improve readability, maintainability, and reduce coupling with other classes.

### Part B

The given class has **low cohesion**. Following Arthur Riel’s heuristics, that requires simple classes that are loosely coupled and each focus on one thing, this class deviates from this by handling **many unrelated responsibilities** such GPA computation, email formatting, payment processing, and file export which extensively violates the *Single Responsibility Principle*. For higher cohesion, the class should be **rearranged into classes with singular focus** such as GPAService for grade calculations, EmailFormatter for messages, and PaymentProcessor for tuition transactions to name a few. This would help each class focus on doing one thing well and make the code more organized and maintainable.

## Question 3

### Part A

The current structure does **not** support changing a car’s trim level because each trim (Base, Sports, Luxury) is a separate subclass of Car. Once you create a Car object, for example new Sports(), it can’t switch to another trim like Luxury without creating a whole new object making the design too rigid. It also mixes too many different ideas in one inheritance chain. If someone later wants both electric and petrol versions of each trim, we would need even more subclasses. So, the design makes it hard to update or change the trim during manufacturing.

### Part B

Define a small Trim interface with getTrimLevel().

Introduce trims: BaseTrim, SportsTrim, LuxuryTrim.

Keep Engine as a separate component (Electric or Petrol).

In Car, let’s have Trim trim and Engine engine. Provide setTrim(Trim t) (and optionally setEngine(Engine e)) which allows them to change at runtime if necessary

**How it solves the problem:**

* Start a build as new Car(new BaseTrim(), new Electric()).
* If the customer changes their mind, call setTrim(new LuxuryTrim()).
* The same Car instance now reflects the new trim. No new object and no class changes favoring composition over inheritance

## Question 5

I have been using ChatGPT to support my learning in this course. It has helped me simplify and better understand questions that I initially found confusing, especially when I wasn’t sure what the problem was asking for. It has also been useful for reviewing my work to ensure that I addressed all the required points and met assignment expectations. It has also encouraged me to think about different ways to approach a problem instead of sticking to my first idea, which has helped me grow in how I analyze and reason through coding tasks.

The main benefits I have found include saving time and being able to get direct, conversational feedback without searching through multiple web resources. I appreciate that I can ask follow-up questions in the same thread to clarify my understanding, which makes the learning process more interactive. One limitation is that sometimes the AI does not fully understand my question, leading to some back-and-forth before I get what I need. Looking ahead, I expect AI to continue improving how I solve problems both academically and professionally by making me more productive and confident. It will also help me find quick answers to complex questions that I might otherwise need to wait for help with, allowing me to work faster and more independently on small picture stuff while focusing more time on big picture things.

## References

ChatGPT transcript - <https://chatgpt.com/share/68f6c9fa-dea4-800f-8142-fb42d3f9a577>

Prompts:

1. Please restate Arthur Riel's heuristics simply again.
2. Please clarify if my github paths attached in the screenshot follow the rubric