

# Discussion Board: Reflection on Software Quality

---

For this week's Discussion Board assignment, I want you to connect your reading about software quality with the programming assignment this week.

As you build your first "meaningful" Java program in a top-down (non-OOP) style, reflect on software quality in a very practical way: What does "quality" look like when you're writing a console program with variables, conditionals, loops, methods, and error handling? Which quality attributes matter most at this stage—readability, correctness, maintainability, testability, usability, performance—and why? Where do you personally feel quality breaks down first in beginner-to-intermediate code (messy input handling, duplicated logic, hard-coded values, giant methods, unclear naming), and what small habits prevent that?

Now connect that to the upcoming refactor: If you had to "future-proof" your non-OOP program knowing it will become object-oriented later, what choices would you make right now to make the transition easier? What parts of your program feel like they should become objects later (e.g., "Account," "Calculation," "Transaction," "Investment Schedule", etc.), and what responsibilities would each one own? What boundaries can you identify between "data," "rules," and "user interaction," and how could you keep those from getting tangled even before you use classes?

End by predicting: what part of your current program will be the hardest to refactor into OOP, and what would make it easier next time you start a project?

Think deeply about software quality and future-proofing.

In addition to writing your Discussion post, ensure you respond to at least two other students in a substantive way.