**LOUIE JAY A. CENIZA APRIL 3, 2024**

**CODE SHOULD BE EASY TO UNDERSTAND (CHAPTER 1)**

**What Makes Code “Better”?.** In this lesson it gives us an insight where you need to have your preferences of factoring your code base on what you want or what makes it better to your understanding.

**The Fundamental Theorem of Readability.** Refers to the guiding principle by emphasizing the importance of minimizing the time it takes for someone to read through and fully understand code. It suggests that this metric, often referred to as "time-till-understanding," is a fundamental aspect of code readability.

**Is Smaller Always Better?.** In this lesson it gives us an insight where smaller code is generally easier to understand, there are situations where clarity and comprehensibility are more important than brevity.

**Does Time-Till-Understanding Conflict with Other Goals?.** In this lesson it answered the question if it conflicts with your goals. Also, the author suggests that while there are other important goals in software development, such as efficiency, architectural design, and testability, these goals do not conflict with prioritizing readability. In fact, code that is easy to understand often leads to better overall code quality, as it tends to be well-architected and easy to test.

**The Hard Part.** In this lesson it tells us that considers others perspective when writing code. The need to actively think about whether someone unfamiliar with the codebase would find it easy to understand. This requires a conscious effort to engage a different mindset—one that prioritizes clarity and comprehensibility over brevity or complexity.