Clean Code

What is clean code 13?

is a book written by Robert C (Uncle Bob), that provides guidelines and principles for writing readable, maintainable, and efficient code.

The book emphasizes the importance of writing code that is easy to understand, The 13 principles of Clean Code are:

- SOLID Principles:

Single Responsibility Principle (SRP): A class should have only one reason to change.

Open/Closed Principle (OCP): Software entities (classes, modules, functions) should be open for extension but closed for modification.

Liskov Substitution Principle (LSP): Subtypes must be substitutable for their base types without affecting program correctness.

Interface Segregation Principle (ISP): Clients should not be forced to depend on interfaces they do not use.

Dependency Inversion Principle (DIP): High-level modules should not depend on low-level modules. Both should depend on abstractions.

- Keep It Simple, Stupid (KISS): Strive for simplicity in design and implementation.
- Don't Repeat Yourself (DRY): Avoid duplication in code.
- Single Responsibility Principle (SRP): A class or module should have only one reason to change.

- Open/Closed Principle (OCP): Software entities should be open for extension but closed for modification.
- Liskov Substitution Principle (LSP): Subtypes must be substitutable for their base types without affecting program correctness.
- Interface Segregation Principle (ISP): Clients should not be forced to depend on interfaces they do not use.
- Dependency Inversion Principle (DIP): High-level modules should not depend on low-level modules, both should depend on abstractions.
- Law of Demeter (LoD): Also known as the "Principle of Least Knowledge." A module should not know about the internal details of the objects it manipulates.
- Favor Composition Over Inheritance: Composition (using objects as building blocks) is often more flexible and maintainable than inheritance.
- Minimize Mutability: Reduce the use of mutable state and side effects. Immutability can lead to more predictable and easier-to-reason-about code.
- Single Level of Abstraction (SLA): Functions and methods should operate at a single level of abstraction.
- Clean Code Naming: Use meaningful and descriptive names for classes, variables, functions, and methods.