**SOLID is an acronym that stands for five basic principles of Object-Oriented Programming**

* Single Responsibility Principle: A class should have a single responsibility. Single Responsibility Principle are inherently cohesive in nature.
* Open/Closed Principle: open for extension, but closed for modification.
* Liskov Substitution Principle: functions that use references to base classes must be able to use objects of the derived class without knowing it.
* Interface Segregation Principle: The Interface Segregation Principle states that clients should not be forced to implement interfaces they don't use. Instead of one fat interface many small interfaces are preferred based on groups of methods, each one serving one submodule.
* Dependency Inversion Principle. High-level modules should not depend on low-level modules. Both should depend on abstractions. Abstractions should not depend on details. Details should depend on abstractions.

YAGNI (You aren't gonna need it) is a principle of extreme programming(XP) that states a programmer should not add functionality until deemed necessary. Always implement things when you actually need them, never when you just foresee that you need them.