SOLID

SRP – Single Responsability Principle

“A class should have, one, and only one, reazon to change”

Em modos mais práticos, uma classe deve ter apenas um motivo para que seja mudada pois ela tem responsabilidade única, ou seja, se ela tiver que sofrer alguma alteração, será referente apenas a essa única responsabilidade existente. O mesmo pode ser aplicado para funções.

OSP - Open Closed Principle

“Software entities (classes, modules, functions...) should be open for extension, but closed for modification.”

Trata-se da ideia de que ao fazemos classes, pensemos na possibilidade de disponibilizar ela apenas para extensão, nunca para modificação. Ou seja, procurar deixar o escopo da classe que está sendo criada bem definido, fazendo com que alterações não sejam necessárias, mas abrindo portas para estensões.

LSP – Liskov Substituiton Principle

“Let q(x) be a property provable about objects x of type T. Then q(y) should be provable for objects y of type S, where S is a subtype of T.”

Em outras palavras, uma classe base deve poder ser substituída por sua classe derivada.

Um exemplo clássico é o do Retângulo e Quadrado. Sabe-se que o quadrado é um tipo de retângulo, portanto, é possível realizar a abstração que a classe “Quadrado” pode estender a classe “Retângulo”, porém isso não deveria ocorrer, já que o retângulo pode ter 2 medidas diferentes (largura e comprimento), enquanto que o quadrado só tem uma medida para todos os seus lados. Isso exemplifica um uso errado do LSP.

ISP – Interface Segregation Principle

“States that no cliente should be forced to depend on methods it does no use”

É mais interessante que existam muitas interfaces pequenas e específicas, do que apenas uma interface.