- In software engineering, SOLID is a mnemonic acronym for five design principles intended to make software designs more understandable, flexible, and maintainable.

- The principles are a subset of many principles promoted by American software engineer and instructor Robert C. Martin, first introduced in his 2000 paper Design Principles and Design Patterns.

- ## The SOLID concepts are : -

1. **The Single-responsibility principle :**
   "There should never be more than one reason for a class to change." In other words, every class should have only one responsibility.

2. **The Open–closed principle :**
   "Software entities ... should be open for extension, but closed for modification."

3. **The Liskov substitution principle :**
   "Functions that use pointers or references to base classes must be able to use objects of derived classes without knowing it."

4. **The Interface segregation principle :**
   "Many client-specific interfaces are better than one general-purpose interface."

5. **The Dependency inversion principle :**
   "Depend upon abstractions, not concretions."

- **The SOLID acronym was introduced later, around 2004, by Michael Feathers.**

- **Although the SOLID principles apply to any object-oriented design, they can also form a core philosophy for methodologies such as agile development or adaptive software development.**

- ## Summary : -

S : Single responsibility principle
O : Open/closed principle
L : Liskov's substitution principle
I : Interface segregation principle
D : Dependency inversion principle