**SOLID Design Principles in Java**

**S – Single Responsibility Principle.**

**O – Open Closed Principle.**

**L – Liskov Substitution Principle.**

**I – Interface Segregation Principle**

**D – Dependency Inversion Principle.**

**S – Single Responsibility Principle.**

Every Class should Have Single Responsibility.

For Example

We have Class called Bank Service and in this we have so many methods like

Deposit, WithDraw, PrintPassbook, getLoanIntrestInfo, SendOTP

**O – Open Closed Principle.**

This Principle says that software entity like classes, modules, functions should be open for extension but close for modification.

**L – Liskov Substitution Principle.**

Objects of a superclass should be replaceable with objects of a subclass without affecting the correctness of the program.

In other words class A is a subtype of class B then we should be able to replace B with A without interrupting the behavior of program.

**I – Interface Segregation Principle**

This principle state that do not force any client to implement an interface which is irrelevant to them.

Clients should not be forced to depend on interfaces they do not use. This means creating smaller, more specific interfaces rather than large, general-purpose ones.

For example if Interface having 5 method and Class A do not want to override all the method than we can’t force to override the method.

**D – Dependency Inversion Principle.**

This principle state that we must use abstraction (abstract classes and interfaces) instead of concrete implementation. High level module should not depend on the low-level module but both should depend on the abstraction.

High-level modules should not depend on low-level modules. Both should depend on abstractions. Also, abstractions should not depend on details. Details should depend on abstractions.