SOLID

S :: Single Responsibility Principle (SRP)

O :: Open-Closed Principle (OCP)

L :: Liskov Substitution Principle (LSP)

I :: Interface Segregation Principle (ISP)

D :: Dependency Inversion Principle (DIP)

SINGLE RESPONSIBILITY PRINCIPLE

→ A class should have only one reason to change ( Class Should only have one responsibility )

OPEN-CLOSED PRINCIPLE

→ A class should be open for extension but closed for modification

class shape ( Should only contain method for all the shapes )

To have a rectangle using shape class, need to make another class named rectangle for only rectangle.

To have a circle using shape class, need to make another class named circle for only circle.

LISKOV SUBSTITUTION PRINCIPLE

→ Subclasses should be substitutable for their base classes without breaking functionality

class Bird ( Should allow all the subclass to implement method in Bird class )

class Penguin ( Penguin cannot fly )

→ place fly() method in a bird that can fly and give penguin different method that connects main class

INTERFACE SEGREGATION PRINCIPLE

→ A class should not be forced to implement interfaces it does not use

Fix unnecessary method when using subclasses

DEPENDENCY INVERSION PRINCIPLE

→ High-level modules should not depend on low-level modules. Both should depend on abstractions

class NotificationService ( Sending any type of message )

All subclasses should have each type of message sendings

( Should not modify other classes to add something )