Taller Principios

SRP – OCP – LCP

**(Single Responsibility Principle / Open Closed Principle / Liskov Principle)**

# Objetivos:

* Identificar incumplimiento a los principios SRP – OCP – LCP.
* Corregir código que incumpla los principios SRP – OCP – LCP.
* Utilizar Github como herramienta colaborativa para cargar los proyectos.
* Lengua de programación JAVA

# Instrucciones

* En taller se debe realizar individualmente.
* Usted trabajará con el proyecto en un repositorio de Github. El proyecto está compuesto por 6 paquetes (2 por cada principio, “Wrong” y “Good”).
* Descargue el proyecto desde el repositorio llamado: https://github.com/neortiz1
* Los paquetes que contienen **‘mal’** en su nombre, contiene clases e interfaces que incumplen un principio. Las tres primeras letras del nombre del paquete le indican el principio que se está incumpliendo.
* Analice el código y reestructure las clases (Refactorizar), de modo que el principio SOLID se esté cumpliendo. Agregue su propuesta de la solución en los sub-paquetes **‘Good’** respectivos.

# Conteste

Explique brevemente como cada principio se incumplió en el código analizado.

**Erick Lasluisa**

* SRP: Se incumplió el principio de Single Responsability al implementar el método reFuel() en la clase Vehicle, ya que este método no tiene cohesión con la clase, es por esto que dicho método fue refactorizado en otra clase que asumimos pertenece a una gasolinera.
* OCP: Se incumplió el principio Open-Closed ya que los modos de manejo (DrivingMode) no están declaradas como clases por lo tanto no están abiertas al reuso y son fácilmente modificables. Por lo tanto, para arreglar este problema iniciamos declarando una clase DrivingMode, la cual será “padre” de las clases Sport, Comfort y Economy, es decir, estas tres ultimas clases heredan de DrivingMode, por lo tanto, este ultimo debe ser de tipo abstracto. Corregido esto se cumple con el principio OCP ya que esta abierto a la extensión y cerrado para modificación.
* LSP: Se incumplió con el principio de Liskov ya que la clase “Padre” Bird hereda las características de Comer y Volar a las clases “Hijas” Duck y Ostrich, sin embargo, solo Duck puede volar, por lo tanto, no está bien aplicado el principio. Para corregir esto se ha creado una clase abstracta FlyingBird la cual hereda de Bird, que fue modificada para que solo tenga de característica comer; en la clase FlyingBird se agrego la característica de volar y posteriormente se declara la clase Duck de forma que hereda de FlyingBird, es decir, tiene las características de comer y volar. Mientras que la clase Ostrich se declaró de forma que hereda de la clase Bird, es decir, solamente puede comer.

# Entregable

* Enlace al repositorio de GitHub donde colocaron su solución a cada principio y este archivo Word.