Taller Principios SOLID

# Objetivos:

* Identificar violaciones a los principios SOLID.
* Corregir código que viole principios SOLID.
* Utilizar Git como herramienta de apoyo al trabajo colaborativo.

# Instrucciones

* Trabajar en los grupos del proyecto.
* Usted trabajará con el proyecto en un repositorio de Github. El proyecto está compuesto por 10 paquetes (2 por cada principio, mal y bien).
* Realizar un Fork al repositorio del grupo, desde el repositorio llamado: <https://github.com/djurado/solid>
* Cada integrante debe haber solucionado por lo menos uno de principios y debe verse reflejado en el usuario de los commits al repositorio.
* Los paquetes que contienen **‘mal’** en su nombre, contiene clases e interfaces que viola un principio SOLID. Las tres primeras letras del nombre del paquete le indican el principio que se está violando.
* Analice el código y reestructure las clases, de modo que el principio SOLID ya no sea violado. Agregue su propuesta de solución en sub-paquetes **‘bien’** respectivos.

# Conteste

Explique brevemente como cada principio es violado en el código analizado.

* SRP: En este punto se violaba el principio, debido a que se tenía un método curar que seteaba nuevamente el máximo valor que podía tener en porcentaje la vida del pokemon, para utilizar el principio cree una nueva clase llamada curacionPokemon la cual tiene un método curar el cual recibe un pokemon y le setea el valor de 100 a su atributo porcentaje de vida.
* OCP:Se violaba el principio, porque en la clase manejar ataque en el método que manejaba estos ataques lo hacía mediante un switch, y esto se lo puede cambiar cambiando la clase atauqe como una interface y luego ir crando clases por cada ataque, y la interface ataque tiene un metodo para clacular el danio de manera que, si queremos agregar un nuevo ataque solo realizamos la nueva clase correspondiente y esta implementaría la interface ataque , y no se tendría que cambiar nada mas.
* LSP:
* ISP: se violaba el principio, porque la interface EntrenamientoPokemon realizaba tres ataques de corte, de fuego y de agua, y según el principio debería existir una interface para cada método entonces cree dos interfaces más que realicen el método AtaqueDeFuego y otra interface que implemente el AtaqueDeAgua.Para reautilizar el código hice que la clase pokemon implemente EntrenamientoPokemon y que las clases hijas implementen una interface según su clase.
* DIP:

# Entregable

* El enlace al repositorio de GitHub donde colocaron su solución a cada principio y este archivo Word con su análisis grupal.
* Subir el enlace a Sidweb.