ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERIA EN ELECTRICIDAD Y COMPUTACION

**Taller de Principios SOLID – Diseño de Software**

# Objetivos:

* Identificar violaciones a los principios SOLID.
* Corregir código que viole principios SOLID.

Una pequeña empresa de postres ha comenzado a armar el sistema de todo su negocio. Los primeros productos que han sacado a la venta son pasteles y helados.

Se sabe también que ya tienen como calcular los precios de cada producto que vende. Los productos tienen un costo parcial fijo, al cual se les aumenta el valor del IVA. A cada postre se le puede añadir distintos aderezos que tienen un costo de 0.50 ctvs. cada uno. Sumando este valor adicional, al anterior se obtiene el valor final de cada producto que venden.

Todos los postres se realizan con leche entera, sin embargo, el negocio ofrece la facilidad de cambiar el tipo de leche a descremada o deslactosada, según el cliente lo prefiera sin ningún costo adicional.

Luego de unos días, una persona que conoce sobre diseño de software ha revisado el sistema, y ha identificado que existen ciertos principios SOLID están siendo violados por el sistema desarrollado.

# Indicaciones

1. Identifique los principios SOLID que no están siendo respetados explicando el porqué.
2. Corrija el código para que solucionar los problemas previamente mencionados, creando un nuevo paquete donde si se respete los principios SOLID.
3. Por cada principio violado, muestre el código que lo incumple y luego muestre y explique como solucionó el problema con su código.

Puede crear clases o interfaces adicionales si considera necesario, pero no elimine el código para poder hacer la comparación final.

# Desarrollo

En el paquete Leche se viola el primer principio (Single-Responsibility Principle) debido al código repetido en el método usar(), se creo una clase abstracta leche que implementa la función usar().

En el paquete postres las clases Helado y Pastel violan el primer principio (Single-Responsibility Principle) debido al código repetido del método calcularPrecioFinal(), se procedio con la creación de la clase CalcularPrecio que contiene al método. Por otro lado viola el segundo principio (Open-Closed Principle) por que tienen clases que tienen las mismas características, por lo tanto se procedio a la creación de la clase postres de la cual heredan Helado y Pastel.

En el paquete Procesos que esta la clase CambioLeche que viola el quinto principio (dependecy Inversion Principle) debido a que depende de dos clases y de sus modificaciones, el cúal no debería suceder, por lo que se procede a implementar el método cambiar() a cada clase de leche existente.