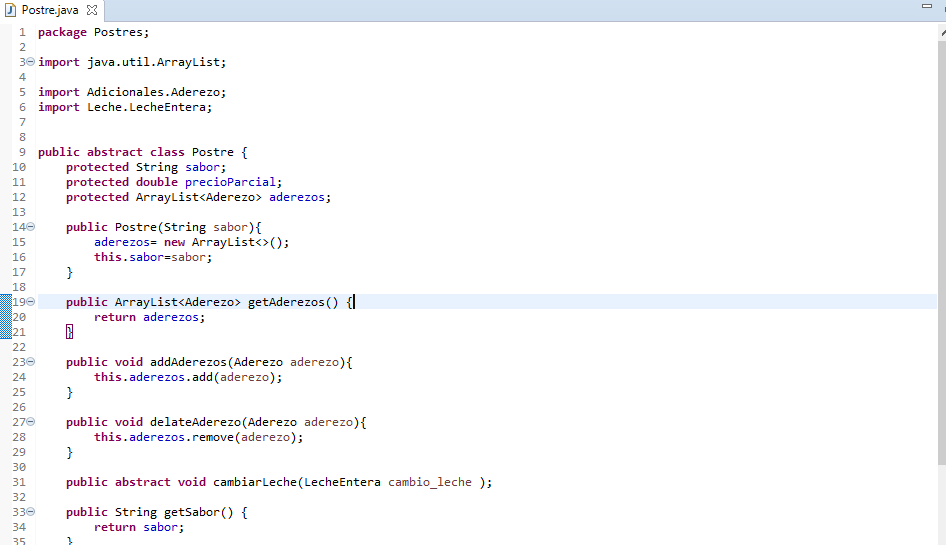
ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

FACULTAD DE INGENIERIA EN ELECTRICIDAD Y COMPUTACION

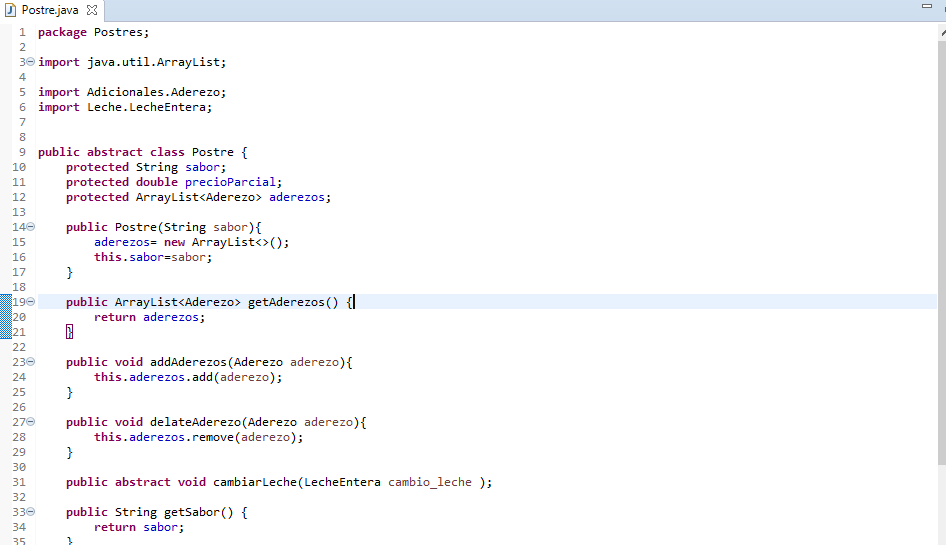
**Taller de Principios SOLID – Diseño de Software**

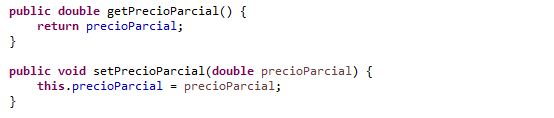
1. **Clases Helado y Pastel. Tienen mucha similitud, se debería crear una clase padre llamada Postre.**

No se está violando ningún principio SOLID ya que solamente se está repitiendo el código en clases diferentes. El problema de la repetición de código se resolvió agregando la clase padre Postre.



1. **Clases Procesos.OperacionesAderezo y Postre. ¿Es necesaria la clase OperacionesAderezo?. Se puede incluir dentro de postre un método para agregar un aderezo y para quitar un aderezo.**

Se está violando el principio de Responsabilidad única ya que se están usando métodos específicos propios de la clase postre en otras clases. No es necesaria la clase OperacionesAderezo por lo que se procedió a eliminar la clase.



1. **Métodos calcularPrecioFinal() y showPrecioFinal() están muy relacionados, deben estar en otra clase por si cambia la fórmula de cálculo. La clase nueva debe llamarse Procesos.ManejadorDePrecio.**
2. **Enum Adicionales.Aderezo es muy estático, debe convertirse en clase abstract con un atributo nombre y un método abstracto setNombre para que cada tipo de aderezo sea una subclase de Aderezo e implemente dicho método. También, sobrescriba el método toString() en la clase Aderezo, para que devuelva el nombre del aderezo en mayúsculas.**
3. **Paquete Leche y la clase Procesos.ManejadorDeLeche. En el main descomente las instrucciones para realizar el cambio del tipo de leche utilizada en cada postre, luego analice como solucionar el error generado en la clase Leche.LecheDeslactosada.**
4. **Cambie el main por el siguiente código y ajuste según su propia implementación.**