Diseño de Software Orientado a Objetos
Práctico 8: Patrón Template Method

Ejercicio 0. Descargue el código provisto en el repositorio git a continuación:

https://github.com/pponzio/oo-design-2024.git

La carpeta `8-template-method` contiene el código complementario a esta práctica. Se proveen scripts gradle para compilar automáticamente el código y ejecutar los tests (ej. ejecutar *gradle test* en línea de comandos). Se recomienda utilizar algún IDE (ej. IntelliJ Idea) para facilitar la codificación, el refactoring, la ejecución de tests y el debugging.

Ejercicio 1. Utilice el código provisto para experimentar con el patrón Template Method. Utilice la metodología de TDD.

a) Implemente nuevas clases que extiendan `CaffeineBeverage` para comprender el funcionamiento del patrón Template Method..
b) Implemente una clase `BlackCoffee` (en el paquete `barista`) que no agregue condimentos y cree un escenario de ejecución que lo incluya.
c) Modifique su diseño para implementar `BlackCoffee` pero incorporando un hook en `CaffeineBeverage` que evite que `addCondiments` sea invocado para `BlackCoffee`. Cree un escenario de ejecución que incluya esta nueva implementación de `BlackCoffee`.

Ejercicio 2. Reimplemente el ejercicio 3 de la práctica 2 (imprimir por pantalla los primeros *n* números primos) pero ahora usando Template Method en lugar de Strategy. Compare los diagramas de clase correspondientes. Explique las diferencias entre las implementaciones y entre estos dos patrones. Utilice la metodología de TDD.

Ejercicio 3. `java.util.AbstractList` implementa un esqueleto (Template Method) para minimizar el esfuerzo requerido para implementar listas. Implemente los métodos requeridos para obtener una implementación de listas inmutables con arreglos, basada en `java.util.AbstractList`. ¿Qué métodos quedan implementados gracias al Template Method? Utilice la metodología de TDD. Provea tests para las funcionalidades de `AbstractList` que no implementó.