Diseño de Software Orientado a Objetos
Práctico 6: Patrón Command

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

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

La carpeta `6-command-remote-control` 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 de las distintas versiones de Remote Control para experimentar y entender el patrón Command. Utilice la metodología de TDD.

a) Agregue un nuevo dispositivo, comandos de encendido y apagado, y cree distintos escenarios de uso asociando el nuevo dispositivo al control remoto y ejecutando sus comandos (paquete `remotecontrol.remote`).
b) Agregue un métodos undo para los comandos del dispositivo creado para el ejercicio anterior, y cree escenarios de uso del dispositivo incluyendo ejecuciones del undo de los comandos (paquete `remotecontrol.remotewithundo`).
c) Cree algunos comandos macro, asócielos al control remoto y pruébelos en diferentes escenarios (paquete `remotecontrol.remotewithundo`).
d) Implemente la funcionalidad de undo para los comandos macro.
e) ¿Tuvo que modificar la implementación de alguna de las clases existentes para resolver los puntos a, b, c y d anteriores?

Ejercicio 2. Cree una versión del control remoto que permita deshacer y rehacer los últimos *k* comandos (para un *k* entero dado). Utilice la metodología de TDD para todo el ejercicio.

Ejercicio 3. Implementar algunos comandos de un editor de texto simple por línea de comandos usando el patrón Command y la metodología TDD.

a) Dado un buffer de texto (para el cual puede elegir la representación que desee) el editor debe permitir cuatro comandos: (1) agregar una línea nueva luego de la i-ésima línea, (2) eliminar la i-ésima línea, (3) imprimir el estado actual del buffer como un texto por pantalla. El usuario ingresa números por línea de comandos para elegir el comando a ejecutar, del 1 al 3 para los comandos anteriores, y 4 y 5 para undo y redo.

b) Cambio en los requisitos: se agregan dos comandos nuevos: agregar una palabra en la i-ésima posición de una línea dada, y eliminar la i-esima palabra. ¿Tuvo que modificar la implementación de alguna de las clases existentes para resolver este problema?