Práctica 3

Equipo: Los Peaky Blinders
Carlos Emilio Castañón Maldonado José Camilo García Ponce Dafne Bonilla Reyes

1 **Menciona los principios de diseño esenciales de los patrones Decorator y Adapter. Menciona una desventaja de cada patrón.**

✶ **Principios de diseño esenciales:**

- **Decorator**

  El patrón de diseño Decorator es un patrón estructural que nos permite agregar dinámicamente funcionalidad y comportamiento a un objeto sin afectar el comportamiento de otros objetos existentes dentro de la misma clase, por lo que no hay cambios en esta clase, digamos, original. Decorator usa herencia para extender el comportamiento de la clase y lo anterior sucede en tiempo de compilación, haciendo que todas las instancias de esa clase obtengan el comportamiento extendido.

  Algo importante de Decorator es que proporciona un envoltorio para la clase existente. Decorator usa clases abstractas o interfaces con la composición para implementar el contenedor. Los patrones de diseño Decorator crean clases de decorador, que envuelven la clase original y brindan funcionalidad adicional al mantener la firma de los métodos de clase sin cambios.

  En conclusión, el patrón de diseño Decorator es útil para proporcionar capacidades de modificación en tiempo de ejecución y, por lo tanto, más flexibles. Es fácil de mantener y ampliar cuando la cantidad de opciones es mayor.

- **Adapter**

  Adapter es un patrón de diseño estructural que permite la colaboración de objetos con interfaces incompatibles.

  El patrón de Adapter convierte la interfaz de una clase en otra interfaz que esperan los clientes, permitiendo que las clases que de otro modo no podrían trabajar juntas debido a las interfaces incompatibles puedan hacerlo.

  Este patrón involucra una sola clase que es responsable de unir funcionalidades de interfaces independientes o incompatibles.

  Usando Adapter tendríamos algo como lo siguiente:

  1. El cliente realiza una solicitud al adaptador llamando a un método mediante la interfaz de destino.
  2. El adaptador traduce esa solicitud en el adaptado usando la interfaz del adaptado.
  3. El cliente recibe los resultados de la llamada y no se da cuenta de la presencia del adaptador.

✶ **Desventajas:**

- **Decorator**

  La principal desventaja de Decorator es la alta complejidad del código. Decorator puede dar lugar a tener un alto número de objetos en el diseño, y el uso excesivo puede volverse engorroso. Además, Decorator puede complicar el proceso de creación de instancias del componente porque no solo tiene que crear una instancia del componente, sino también envolverlo en varios decoradores.

  Asimismo, este patrón de diseño también tiende a complicar que los decoradores realicen un seguimiento de otros decoradores, ya que mirar hacia atrás en múltiples capas de la cadena del decorador comienza a empujar al patrón del decorador más allá de su verdadera intención complicando la depuración del código.

- **Adapter**

  Algunas veces, se requieren muchas adaptaciones a lo largo de una cadena adaptadora para alcanzar el tipo que se requiere, haciendo que sea más fácil re-codificar manualmente la interfaz del cliente que usar el patrón Adapter, sin contar que usando este patrón de diseño, también aumentamos la complejidad general del código.

2 Uso del programa

- Compilar desde `src/`

```
javac *.java PizzasDonCangrejo/*.java WaySub/*.java
```

- Ejecutar desde `src/`

```
java Practica3
```

- Generar la documentación desde `src/`

```
javadoc -d docs *.java PizzasDonCangrejo/*.java WaySub/*.java
```

➢ **Explicación:**

Para iniciar el programa, primero es necesario compilar y ejecutar el programa. Después, los pasos para ordenar se mostrarán en terminal.

Si se quiere generar la documentación, esto sería con el comando dado arriba, y luego, los archivos se generarán en el directorio llamado *docs*.

3 Implementación

Para poder facilitar la implementación de los sandwiches, elegimos usar el patrón de Decorator.

A continuación, en la parte de las pizzas y su unión a sandwiches, usamos el patrón Adapter para así poder tratarlas como sandwiches.