PRACTICA 2

Equipo: *Los timidos*

Integrantes:

Nombre	*No. de cuenta*
Castillo Camacho Violeta Ardeni	*318214328*
Fernández Blancas Melissa Lizbeth	*319281778*
Sánchez Salmerón Ethan Damián	*319122323*

SECCION TEORICA:

1. Menciona los principios de diseño esenciales del patrón State, Template e Iterator.
Menciona una desventaja de cada patrón.

STATE:

- Se utiliza el patrón State cuando se tenga un objeto que se comporta de forma diferente dependiendo de su estado actual, también si el número de estados es muy grande o si algun código de un estado en específico cambie con frecuencia.

- Este patrón nos ayuda a extraer todo el código del estado en específico y lo podamos meter en un grupo de clases específicas, así, modificar, agregar o eliminar estados no resultará tan pesado de cambiar.

-Se puede usar State cuando tengamos una clase llena de métodos que contienen muchas condicionales, donde su comportamiento cambia si se cumplen ciertos criteros.

-Organiza el código relacionado con estados particulares en clases separadas, esto facilita poder desarrollar las clases como diagramas de estados, sin meter tantas condicionales al código.

Desventajas:

- Si se tiene un objeto cuyos estados apenas varian, implementar State resultará excesivo

- No se puede suar State si los estados son muy diferentes entre sí.

- El objeto en contexto debe saber que tipos de estados hay, el no hacerlo resultaría en no poder realizar un diagrama de estados facil de modificar.

TEMPLATE:

- Se usa Template cuando se quiera extender pasos particulares de un algoritmo, así, no se altera el comportamiento de la superclase, y sus clases hijas son las que deben implementar esos pasos especiales.

- Cuando tengamos varias clases con comportamientos muy similares, podemos usar Template. Crear una clase con el comprotamiento general, y dejar metodos abstractos para que sus clases hijas puedan implementarlos a su gusto.

- Si usamos Template quitamos el dublicado de código, al abstraer ciertos comportamientos a una clase, que sabemos que todas sus clases hijas tendrán.

Desventajas:

- La clase esqueleto puede resultar pesada de implementar si el número de metodos abstractos es muy grande. Es posible que ni siqueira tenga sentido usar una clase abstracta.

- Quien quiera usar el esqueleto de la clase padre, se verá limitado a lo que puede o no hacer las clases que quiera implementar.

ITERATOR:

- Cuando se usa Iterator, es porque tenemos una coleccion de objetos con una estructura de datos lo suficientemente compleja como para ocultar su uso a las personas que usarán el producto.

- Ayuda a no duplicar código. El hacer algortimos que sean capaces de iterar en las estructuras de datos, hace nuestro cpodigo más limpio y breve.

- El hacer que las clases implemeten ciertas interfaces para recorrer las estructuras de datos, resultará más facil poder agregar más clases también contengan estructuras de datos donde sea dificl acceder a los objetos de esa estructura.

Desventajas:

- No resulta conveniente usar Iterator si nuestras clases contienen estructras de datos sencillas.

- A veces un Iterador resulta menos eficiente que recorrer los elementos directamente de algunas colecciones que tengan métodos que ayuden a hacer esta tarea mas fácil.

- Si la colección cambia en tiempo de ejecución, el iterador podría verse afectado.

NOTAS DEL CODIGO:

- La clase que contiene el main principal es el archivo que lleva con nombre Main.java
- Se compila y ejecuta de la forma tradicional (con javac y java).