

1. Vocabulario y conceptos

1. ¿Por qué debe aplicar un patrón de diseño siempre que sea posible?

Solución COMPLETAR.

2. Enuncie una posible desventaja al usar un patrón de diseño.

Solución COMPLETAR.

3. Explique en qué fase del ciclo de producción de un sistema de software usaría los patrones de diseño.

Solución COMPLETAR.

4. Aclare la o las razones por las cuales seleccionaría el patrón Strategy en lugar de Visitor en el caso en que la función asignada a ambos deba recorrer, al menos, cierta porción de una estructura de objetos.

Solución COMPLETAR.

5. Describa la forma en que es conveniente seleccionar un patrón de diseño.

Solución COMPLETAR.

6. Explique la diferencia entre herencia de clases y herencia de tipos o interfaces. ¿Cuál es la que más conviene usar desde el punto de vista del diseño? ¿Por qué?

Solución COMPLETAR.

2. Aplicación de patrones de diseño

En todos los problemas que siguen se debe aplicar el o los patrones de diseño indicados para generar un DOO. Cada diseño debe documentarse como se explicó en clase y como se pide en la práctica de diseño.

1. Aplicar el patrón Command al problema «ESTACIÓN CLIMATOMÉTRICA» descrito en la práctica de diseño.

Solución

Pattern based on	Nombre Command	
because	<ul style="list-style-type: none"> ■ Facilita añadir nuevas características al sistema. Por ejemplo se puede crear una nueva orden que emita alarmas según los valores. ■ Desacopla el temporizador de los sensores y el repositorio. 	
where	Invocador	is Temporizador
	OrdenConcreta	is Muestrador
	Ejecutar()	is Ejecutar()
	Receptor	is Sensores
	Accion()	is Sensor()
	Receptor	is Repositorio
	Accion()	is AgregarMediciones()
	OrdenConcreta	is Calculador
	Ejecutar()	is Ejecutar()
	Receptor	is Repositorio
	Accion()	is AgregarMediciones()
comments	<ul style="list-style-type: none"> ■ El repositorio almacena el historial de mediciones, máximos y mínimos. ■ El muestrador debe guardar los datos en el repositorio luego de sensar. ■ Sensar y AgregarMediciones son en realidad varias acciones. 	

2. Aplicar el patrón Command al problema «ACCESO A UN CANAL DE NAVEGACIÓN» descrito en la práctica de diseño.

Solución COMPLETAR.

3. Aplicar el patrón Composite al problema «MANTENIMIENTO DE CATÁLOGOS» descrito en la práctica de diseño. Hacerlo para definir con detalle la estructura de los catálogos. Cada producto ofertado en un catálogo tiene las siguientes características: nombre, código, descripción, varios niveles de secciones (por ejemplo, Ferretería, Tornillos y Bulones, Tornillos, Tornillos de acero, Tornillos de 1,5mm, etc.), precio, marca, color, etc.

Solución COMPLETAR.

4. Respecto del problema 3, aplicar el patrón Iterator para recorrer la estructura de un catálogo.

Solución COMPLETAR.

5. Defina operaciones de búsqueda por precio, código, sección y palabra en la descripción aplicando el patrón Visitor sobre cada catálogo del problema 3. Muestre código de ejemplo de cómo se implementaría una de las búsquedas.

Solución COMPLETAR.

6. Use el patrón Iterator para muestrear los valores sensados en el problema 1.

Solución

Pattern based on	Nombre Iterator	
because	<ul style="list-style-type: none"> ■ Permite varios recorridos sobre los sensores. ■ Oculta la estructura donde se almacenan los sensores. ■ Define una interfaz uniforme. 	
where	Cliente crearIterador() AgregadoConcreto primero() siguiente() haTerminado() elementoActual() IteradorConcreto IteradorConcreto	is Muestrador is crearIterador() is Sensores is primero() is siguiente() is hayMas?() is sensor() is AccesoAlfabetico is AccesoAleatorio
comments	El muestrador utilizará un iterador para tomar cada muestra y poder actualizar el repositorio.	

7. La empresa de venta por correo desea poner sus catálogos en su sitio Web. Además desea que los catálogos puedan mostrarse ordenados de diferente forma. Por ejemplo, orden alfabético de productos, de secciones de nivel 1, por precio, etc. Aplique el patrón Strategy para implementar esta funcionalidad.

Solución COMPLETAR.

8. Considere el problema del banco de la práctica de diseño. Ahora suponga que el banco ha comprado otros bancos. Cada banco comprado tiene su propia forma de definir una cuenta (caja de ahorro, cuenta corriente, en pesos y dólares), plazo fijo, cliente, etc. Estudie las posibilidades de aplicar (y eventualmente aplique) el patrón Abstract Factory para que el sistema compuesto cree los diferentes productos.

Solución COMPLETAR.

9. Continuando con el problema de la estación climatométrica, suponga que la empresa que la desarrolla tiene varios modelos diferentes con sensores de distinta precisión, robustez y precio. Utilice el patrón Bridge para diseñar un sistema que se auto configure en tiempo de ejecución de acuerdo a la plataforma donde está ejecutando. Muestre código de ejemplo de cómo se implementaría este aspecto de la aplicación.

Solución COMPLETAR.

10. En el problema 8 se trató el problema de la creación de objetos que representan los diversos productos de los bancos que fue adquiriendo nuestro banco. Ahora toca el turno de aplicar el patrón Wrapper para implementar resoluciones del BCRA referidas a las transacciones que pueden realizar los clientes de cualquier banco sobre sus cuentas corrientes. Algunas de las resoluciones del BCRA son las siguientes:

A-2156. No se podrán realizar más de dos extracciones semanales de las cajas de ahorro en pesos.

A-3401. Si el titular es una persona jurídica cada depósito de más de \$10.000 deberá ser informado al BCRA en un archivo mensual descrito en la C-BCRA-B-1190.

A-3436. Los plazos fijos de más de \$1.000.000 deberán ser informados al BCRA en un archivo mensual descrito en la C-BCRA-B-1284.

B-3441. Los bancos quedan autorizado a no permitir extracciones de más de \$10.000 por mes de cualquier cuenta, si así lo consideran apropiado. Esta medida podrá ser aplicada y revocada por cada entidad si lo comunica a sus cliente con una antelación superior a los 30 días.

A-3211. Si un cliente posee valores por más de \$10.000.000 deberá ser informado mensualmente al BCRA.

Solución COMPLETAR.

11. Considere el problema relativo al sistema de archivos de Linux visto en la práctica de diseño. El VFS da una representación jerárquica a los archivos y directorios como usted sabe. Utilice el patrón Composite para representar los directorios y archivos.

Solución COMPLETAR.

12. Aplique el patrón Iterator para recorrer el sistema de archivos de diferentes formas.

Solución COMPLETAR.

13. Suponga que el VFS provee un módulo que dado un código de error emitido por alguna de sus subrutinas (llamadas al sistema) retorna el mensaje correspondiente. Aplique el patrón Bridge para poder configurar el VFS para retornar los errores en diferentes idiomas (internacionalización).

Solución COMPLETAR.

14. Una utilidad provista por Linux es *du* que retorna el espacio en disco utilizado por un directorio dado. Estas utilidades se proveen a nivel de usuario. Analice la posibilidad de aplicar (y eventualmente aplique) el patrón Visitor para implementar este tipo de servicios (*find*, etc.).

Solución COMPLETAR.

15. Otra utilidad provista por Linux es el comando *fscheck* que permite reparar el sistema de archivos ante algunos daños. Analice la posibilidad de aplicar (y eventualmente aplique) el patrón Strategy para implementar diferentes estrategias de reparación. Haga lo mismo con el patrón Visitor. Notar que se requiere recorrer el sistema de archivos completo.

Solución COMPLETAR.

16. Suponga que se desean implementar diferentes modelos de control de acceso a los archivos y directorios del sistema de archivos de Linux. Analice la posibilidad de aplicar (y eventualmente aplique) el patrón Wrapper para implementar esos modelos de seguridad.

Solución COMPLETAR.