DISEÑO DE SISTEMAS 2025

. ¿Cuál es el objetivo principal del patrón de diseño Bridge?

A. Permitir crear objetos complejos paso a paso
B. Separar una interfaz de su implementación para que ambas puedan evolucionar de forma independiente
C. Modificar el comportamiento de un objeto en tiempo de ejecución
D. Garantizar que una clase tenga solo una instancia

---

2. ¿Cuál de los siguientes es un patrón de comportamiento?

A. Abstract Factory
B. Prototype
C. Interpreter
D. Decorator

---

3. ¿Qué patrón se utiliza para simplificar el uso de un sistema complejo, proporcionando una interfaz unificada?

A. Adapter
B. Mediator
C. Facade
D. Composite

---

4. ¿Qué patrón permite añadir funcionalidad a un objeto en tiempo de ejecución sin alterar su clase?

A. Observer
B. Strategy
C. Decorator
D. Proxy

---

5. En el patrón Flyweight, ¿qué tipo de información se comparte entre objetos para optimizar recursos?

A. Comportamiento dinámico
B. Estado externo
C. Estado interno inmutable (intrínseco)
D. Funcionalidad encapsulada

---

6. ¿Cuál es la diferencia fundamental entre los patrones Adapter y Decorator?

A. Adapter modifica la estructura, Decorator modifica la interfaz
B. Adapter convierte interfaces incompatibles, Decorator agrega responsabilidades
C. Adapter se usa solo en tiempo de ejecución
D. Decorator requiere herencia múltiple

---

7. ¿Para qué se utiliza el patrón Mediator?

A. Para evitar la creación de múltiples instancias
B. Para centralizar la comunicación entre objetos y reducir el acoplamiento
C. Para crear objetos de manera flexible
D. Para encapsular el estado de un objeto

---

8. ¿Qué patrón encapsula una familia de algoritmos para que puedan ser intercambiados sin modificar el cliente?

A. Visitor
B. Strategy
C. State
D. Builder

---

9. ¿Qué problema resuelve el patrón Memento?

A. Permite clonar objetos sin conocer su clase
B. Guarda el estado interno de un objeto sin romper el encapsulamiento
C. Establece comunicación entre múltiples objetos
D. Asigna responsabilidades entre objetos relacionados

---

10. ¿Qué característica define al patrón Chain of Responsibility?

A. Ejecuta operaciones en paralelo
B. Permite que múltiples objetos puedan procesar una solicitud sin que el emisor sepa cuál lo hará
C. Controla la cantidad de instancias
D. Crea familias de objetos relacionados

---

11. ¿Cuál es el propósito del patrón Command?

A. Encapsular una petición como un objeto para permitir su almacenamiento, ejecución y deshacer
B. Delegar el control a clases hijas
C. Reducir el acoplamiento entre cliente y servidor
D. Sincronizar objetos en una cadena

---

12. ¿Qué patrón se emplea para construir un objeto complejo paso a paso?

A. Prototype
B. Factory Method
C. Builder
D. Singleton

---

13. ¿Por qué se utiliza el patrón Abstract Factory?

A. Para crear instancias únicas
B. Para clonar objetos complejos
C. Para crear familias de objetos relacionados sin especificar sus clases concretas
D. Para establecer dependencias entre clases

---

14. ¿Qué patrón permite cambiar el comportamiento de un objeto sin modificar su código fuente ni usar herencia?

A. Strategy
B. Adapter
C. State
D. Builder

---

15. ¿Cuál es una consecuencia negativa del uso excesivo del patrón Singleton?

A. Aumenta la cohesión del sistema
B. Mejora la escalabilidad
C. Introduce acoplamiento global y dificulta las pruebas
D. Reduce el uso de memoria

---

16. ¿Cuál es la diferencia entre los patrones Prototype y Builder?

A. Prototype crea objetos paso a paso; Builder los clona
B. Prototype clona objetos existentes; Builder los construye paso a paso
C. Builder y Prototype son equivalentes
D. Builder solo se aplica a estructuras recursivas

---

17. En el patrón Visitor, ¿qué hace el "visitante"?

A. Invoca métodos de comunicación entre clases
B. Representa una acción que se aplica a elementos de una estructura
C. Controla el acceso a una clase
D. Encapsula una petición como objeto

---

18. ¿Qué patrón permite tratar de forma uniforme a objetos individuales y compuestos en una jerarquía?

A. Proxy
B. Composite
C. Facade
D. Bridge

---

19. ¿Qué patrón se utiliza cuando se necesita un objeto intermediario que controle el acceso a otro objeto real?

A. Proxy
B. Adapter
C. Factory
D. Observer

---

20. ¿Qué función cumple el patrón Observer?

A. Permite crear objetos sin acoplarse a sus clases concretas
B. Define una relación de dependencia de uno a muchos para notificar cambios automáticamente
C. Controla el acceso a un recurso
D. Almacena y recupera el estado de un objeto