Tema 9

Clases Abstractas:

Son clases que no se pueden instanciar directly (no puedes crear objetos de ellas).
Se usan para definir un tipo genérico o agrupar clases relacionadas.
Pueden contener código reutilizable para las subclases.
Se definen con la palabra clave abstract.
Pueden tener métodos abstractos y concretos.

Métodos Abstractos:

Son métodos declarados pero no implementados en la clase abstracta.
Las subclases deben implementar los métodos abstractos (a menos que la subclase también sea abstracta).

- Se declaran con la palabra clave abstract.
- No tienen cuerpo (la implementación va en la subclase).
- Interfaces:
- Son como "contratos" que definen un conjunto de métodos que una clase debe implementar.
- Una clase puede implementar múltiples interfaces.
- No pueden tener implementación de métodos (hasta Java 8 con los métodos default y static).
- Todos los métodos en una interfaz son implícitamente public y abstract.

Interfaces vs. Clases Abstractas:

Interfaces:

- No pueden implementar métodos (antes de Java 8).
- Una clase puede implementar múltiples interfaces.
- No forman parte de la jerarquía de clases.
- Definen un "contrato" de métodos a implementar.

Clases Abstractas:

- Pueden tener métodos implementados (código reutilizable).
- Una clase solo puede heredar de una clase abstracta.
- Forman parte de la jerarquía de clases.
- Definen un "esqueleto" de clase con métodos a implementar y reutilizar.

Cuándo usar qué:

- **Clase normal (no abstracta):** Cuando la clase representa un objeto concreto que se puede instanciar.
- Subclase (herencia): Cuando se necesita especializar una clase existente (añadir o modificar funcionalidad).
- **Clase Abstracta:** Cuando se define un grupo genérico de clases, se quiere reutilizar código, o se fuerza una API (conjunto de métodos) en las subclases, y no se quiere que la clase base se instancie.
- **Interface:** Cuando se define un grupo genérico de clases SIN código reutilizable, o cuando una clase necesita "ser" múltiples cosas (implementar múltiples "contratos").

¿Cuál de las siguientes afirmaciones es CIERTA sobre las clases abstractas?
a) Se pueden instanciar directamente.
b) No pueden contener métodos.
c) Se definen con la palabra clave abstract.
d) Todas las subclases deben ser abstractas.

Un método abstracto en Java...
a) Tiene una implementación completa en la clase abstracta.
b) No tiene cuerpo y debe ser implementado por las subclases.
c) Se declara con la palabra clave final.
d) Puede ser privado.

¿Cuál es la palabra clave utilizada para definir una clase abstracta en Java?
a) interface
b) extends
c) abstract
d) implements

¿Puede una clase abstracta contener métodos concretos (con implementación)?
a) Sí
b) No

Si una clase hereda de una clase abstracta, ¿qué debe hacer con los métodos abstractos de la superclase?
a) Ignorarlos.
b) Implementarlos.
c) Declararlos como final.
d) Hacerlos privados.

¿Cuál de las siguientes afirmaciones es FALSA sobre las interfaces en Java (antes de Java 8)?
a) Definen un conjunto de métodos que una clase debe implementar.
b) Pueden tener métodos con implementación.
c) Una clase puede implementar múltiples interfaces.
d) Todos sus métodos son implícitamente public y abstract.

¿Cuál es la palabra clave utilizada para implementar una interfaz en Java?
a) extends
b) abstract
c) interface
d) implements

¿Cuántas interfaces puede implementar una clase en Java?
a) Solo una.
b) Solo dos.
c) Un número ilimitado.
d) Ninguna.

¿Puede una interfaz heredar de otra interfaz?
a) Sí
b) No

¿Cuál de las siguientes NO es una diferencia entre una interfaz y una clase abstracta?
a) Una clase puede implementar múltiples interfaces, pero solo heredar de una clase.
b) Una interfaz no puede tener métodos con implementación (antes de Java 8).
c) Una clase abstracta no puede tener métodos abstractos.
d) Las interfaces no forman parte de la jerarquía de clases.

¿Cuándo es más apropiado usar una clase abstracta en lugar de una interfaz?
a) Cuando se quiere definir un "contrato" de métodos a implementar.
b) Cuando se quiere permitir que una clase "sea" múltiples cosas.
c) Cuando se quiere proporcionar código reutilizable para las subclases.
d) Cuando no se quiere que la clase base se instancie.

¿Cuándo es más apropiado usar una interfaz en lugar de una clase abstracta?
a) Cuando se quiere definir un grupo genérico de clases con código reutilizable.
b) Cuando una clase necesita heredar de otra clase.
c) Cuando se quiere definir un "contrato" de métodos a implementar sin proporcionar implementación.
d) Cuando no se quiere que la clase base se instancie.

¿Qué ocurre si una subclase de una clase abstracta no implementa todos los métodos abstractos de la superclase?
a) La subclase debe ser declarada como final.
b) La subclase debe ser declarada como abstract.
c) Se produce un error en tiempo de ejecución.
d) Los métodos abstractos se heredan sin cambios.

¿Cuál es el propósito principal de una clase abstracta?
a) Definir objetos concretos.
b) Forzar una API (conjunto de métodos) a las subclases.
c) Implementar todos los métodos de una interfaz.
d) Ninguna de las anteriores.

¿Cuál es el propósito principal de una interfaz?
a) Proporcionar código reutilizable.
b) Definir una jerarquía de clases.
c) Especificar un conjunto de métodos que las clases que la implementan deben tener.
d) Evitar la instanciación de clases.

¿Puede una clase abstracta tener constructores?
a) Sí
b) No
Si una interfaz declara un método void miMetodo(), ¿cómo debe ser implementado en una clase que implementa la interfaz?
a) private void miMetodo() {}
b) protected void miMetodo() {}
c) public void miMetodo() {}
d) void miMetodo() {}

¿Qué significa que una clase "implementa" una interfaz?
a) Que la clase hereda de la interfaz.
b) Que la clase tiene una relación "es-un" con la interfaz.
c) Que la clase proporciona una implementación para todos los métodos declarados en la interfaz.
d) Que la clase no puede heredar de ninguna otra clase.

¿Cuál de las siguientes afirmaciones es CIERTA sobre la herencia en Java?
a) Una clase puede heredar de múltiples clases.
b) Una clase abstracta no puede heredar de otra clase.
c) Una clase puede heredar de una interfaz.
d) Una clase puede heredar de una sola clase.

¿Qué tipo de relación se establece entre una clase y una interfaz que implementa?
a) "Tiene-un"
b) "Es-un"
c) "Usa-un"
d) "Depende-de"

¿Cuál de las siguientes opciones describe mejor el concepto de "API" en el contexto de clases abstractas e interfaces?
a) Una interfaz de usuario.
b) Un conjunto de reglas y especificaciones que permiten a diferentes componentes de software comunicarse.
c) Un tipo de base de datos.
d) Un lenguaje de programación.

¿Por qué no se pueden instanciar las clases abstractas?
a) Porque no tienen constructores.
b) Porque sus métodos no tienen implementación.
c) Porque están diseñadas para ser generalizaciones que se completan en las subclases.
d) Porque solo pueden contener variables estáticas.

¿Qué sucede si una clase abstracta contiene solo métodos concretos?
a) Es ilegal en Java.
b) Se puede instanciar normalmente.
c) Aún no se puede instanciar, pero puede proporcionar funcionalidad reutilizable.
d) Se convierte automáticamente en una interfaz.

¿Cuál es la principal ventaja de usar interfaces en el diseño de software?
a) Permite la herencia múltiple de implementación.
b) Promueve el acoplamiento débil y la flexibilidad.
c) Aumenta la velocidad de ejecución del programa.
d) Simplifica la gestión de la memoria.

¿Cuál de las siguientes NO es una característica de las interfaces en Java?
a) Pueden contener métodos estáticos (desde Java 8).
b) Pueden contener métodos default (desde Java 8).
c) Pueden contener campos de instancia.
d) Definen un contrato para las clases que las implementan.

1. c
2. b
3. c
4. a
5. b
6. b
7. d
8. c
9. a
10. c
11. c y d (ambas son correctas en diferentes contextos)
12. c
13. b
14. b
15. c
16. a
17. c
18. c
19. d
20. b
21. b
22. c
23. c
24. b
25. c