Ejercicios propuestos

1. Describe la diferencia entre métodos finales y métodos abstractos

Los métodos finales y los métodos abstractos son dos conceptos opuestos en la programación orientada a objetos. Un método final es un método que no se puede sobrescribir ni redefinir por una subclase, mientras que un método abstracto es un método que no tiene implementación y debe ser sobrescrito y definido por una subclase.

2. Explica la diferencia entre métodos sobrecargados y métodos sobrescritos.

Los métodos sobrecargados y los métodos sobrescritos son dos formas de definir métodos con el mismo nombre en una clase, pero con diferentes comportamientos. La diferencia principal es que la sobrecarga ocurre cuando se definen métodos con el mismo nombre pero con diferentes tipos o número de parámetros, mientras que la sobreescritura ocurre cuando se redefine el comportamiento de un método heredado de una clase padre o de una interfaz.
Se recomienda usar la anotación @Override para indicar que se está sobrescribiendo un método, lo que ayuda a evitar errores de compilación o de lógica.

3. ¿Es posible acceder a los campos de una superclase desde una subclase? Razona tu respuesta

Sí, es posible acceder a los campos de una superclase desde una subclase, siempre que dichos campos no sean privados. Los campos privados sólo son accesibles dentro de la misma clase donde se definen, y no se pueden heredar ni modificar por las subclases.
Para acceder a los campos de una superclase desde una subclase, se pueden usar las siguientes opciones:

- Usar directamente el nombre del campo, si el campo es público o tiene el modificador de acceso por defecto (sin especificar) y la subclase está en el mismo paquete que la superclase.

- Usar un método getter o setter de la superclase, si el método es público o protegido, o tiene el modificador de acceso por defecto y la subclase está en el mismo paquete que la superclase.

- Usar la palabra clave super seguida de un punto y el nombre del campo o del método, si el campo o el método no son privados. Esta opción permite acceder al campo o al método de la superclase incluso si la subclase tiene un campo o un método con el mismo nombre.

4. ¿Pueda Java implementar la herencia múltiple? Razona tu respuesta

La herencia múltiple es la capacidad de una clase de heredar de más de una clase padre. Java no permite la herencia múltiple de clases, pero sí la herencia múltiple de interfaces

5. ¿En qué consiste la ejecución polimorfa? Propón un ejemplo

La ejecución polimorfa es la capacidad de un lenguaje de programación orientado a objetos de invocar el método adecuado de una clase o de una de sus subclases en tiempo de ejecución, según el tipo real del objeto que recibe el mensaje. Esto permite que un mismo método tenga diferentes comportamientos según el objeto que lo implementa, lo que facilita la reutilización y la adaptación del código.

6. ¿Qué es una interfaz? Desarrolla tu respuesta

Una interfaz en programación es un conjunto de métodos, propiedades o eventos que definen el comportamiento o la funcionalidad de una clase o de un componente. Una interfaz no contiene la implementación de los métodos, sino solo su declaración, es decir, su nombre, tipo de retorno y parámetros. Una interfaz se puede implementar por una o más clases, que deben proporcionar el código de los métodos declarados en la interfaz. De esta forma, se establece un contrato entre la interfaz y las clases que la implementan, garantizando que estas últimas cumplen con los requisitos de la interfaz.
Las interfaces se usan para lograr la abstracción, el polimorfismo y la adaptabilidad en la programación orientada a objetos.

7. Si se desea manejar información sobre, únicamente, las figuras geométricas punto, recta y circunferencia, ¿qué tipo de relación establecerías entre ellas? Justifica tu respuesta

Sería algo parecido al ejemplo del ejercicio 5 que he adjuntado, mediante una clase abstracta llamada Figura, que contenga los atributos y métodos comunes a todas las figuras geométricas, como el color, el grosor, el área, el perímetro, etc. Luego, se podrían definir tres subclases que hereden de Figura: Punto, Recta y Circunferencia. Cada una de estas subclases tendría sus propios atributos y métodos específicos, como las coordenadas, la pendiente, el radio, etc. Además, se podrían sobrescribir o redefinir algunos de los métodos heredados de Figura, como el área o el perímetro, para adaptarlos a cada tipo de figura

8. Haz un resumen de las palabra reservas que emplean en el manejo de la herencia

super: Esta palabra se usa para referirse a la superclase o a la clase padre de una subclase

extends: Esta palabra se usa para indicar que una clase hereda de otra clase.

abstract: Esta palabra se usa para declarar una clase o un método abstracto. Una clase abstracta es una clase que no se puede instanciar y que contiene al menos un método abstracto.

@Override: Esta palabra se usa para anotar un método que sobrescribe o redefine el comportamiento de un método heredado de la superclase o de una interfaz

9. Si A es una clase de la cual hereda B y B es una clase de la cual hereda C, ¿es posible asignar a una variable declarada de tipo A un objeto de la clase C?

Sí, es posible asignar a una variable declarada de tipo A un objeto de la clase C, pero no es posible asignar a una variable declarada de tipo C un objeto de la clase A. Esto se debe al concepto de polimorfismo, que permite que una variable de un tipo padre pueda referenciar a un objeto de un tipo hijo, pero no al revés.

¿Es posible asignar a una variable declarada de tipo C un objeto de la clase A?

Una variable de tipo C no puede referenciar a un objeto de tipo A, porque A no tiene los atributos y métodos específicos de C2.