**[Módulo - 21] Polimorfismo:**

**(123) -** ¿Qué es el **polimorfismo** en el contexto de la programación orientada a objetos?

•Es la propiedad que tienen los objetos de permitir invocar genéricamente un comportamiento (método) cuya implementación será delegada al objeto correspondiente recién en tiempo de ejecución.

•En otras palabras, es la capacidad de tratar objetos diferentes de la misma forma.

•El polimorfismo tiende a existir en las relaciones de herencia.

**(124) -** ¿Qué implica el polimorfismo basado en herencia?

El polimorfismo implica la definición de métodos y/o propiedades en una clase base, mediante el uso de la palabra reservada virtual y sobrescribirlos con nuevas implementaciones en clases derivadas con la palabra reservada override.

**[Módulo - 22] Sobreescritura de métodos (Override):**

**(125) -** ¿Dónde reside la definición del método a sobrescribir? ¿Qué palabra clave se usa para definirlo?

La definición del método reside en la clase base.

•Se utiliza virtual , o abstract.

**(126) -** ¿Dónde reside la implementación del método a sobrescribir? ¿Qué palabra clave se usa para implementarlo?

La implementación del método reside en la clase derivada.

•Se utiliza override.

**(127) -** ¿Cuándo se resuelve la invocación? (Tiempo de ejecución o compilación)

La invocación es resuelta en tiempo de ejecución.

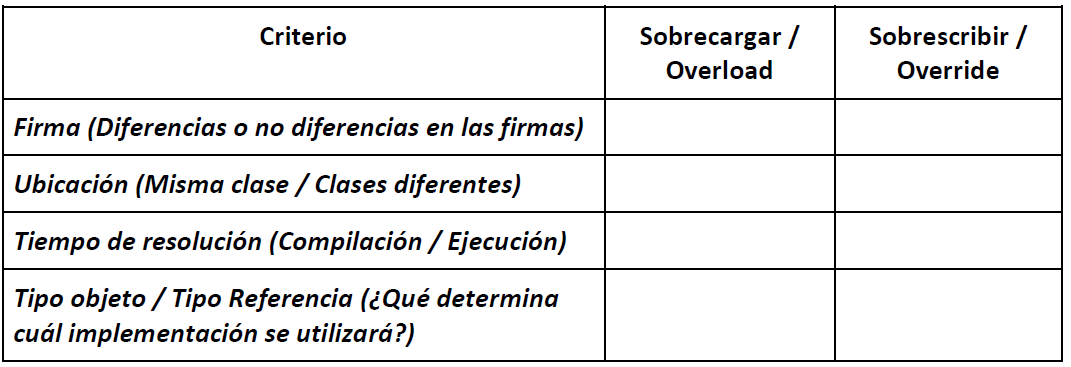
**(128) -** ¿Cuáles son las diferencias entre **sobrecargar (overload) y sobrescribir (override)** un método? (Llenar la tabla)

A dif igual

B misma distinta

C compilacion ejecucion

D referencia(parametro) - objeto



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:::::\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**[Módulo - 23] Clases y miembros abstractos:**

**(129) -** Si quiero declarar un método que pueda ser sobrescrito en las clases derivadas, ¿qué modificador debo usar?

virtual tipo Nombre {

codigo

};

**(130) -** Si quiero declarar un método que deba ser sobrescrito en las clases derivadas, ¿qué modificador debo usar?

Abstract tipo Nombre ();

**(131) -** ¿Qué es una **clase abstracta**? ¿Cuál es su función?

Es una clase que DEBE ser heredada , se utiliza para jerarquizar el código y encapsularlo.

No se puede crear una instancia de una clase abstracta directamente, y es un error en tiempo de compilación utilizar el operador new en una clase abstracta.

•Aunque es posible tener variables y valores cuyos tipos en tiempo de compilación sean abstractos, tales variables y valores serán null contendrán referencias a instancias de clases no abstractas derivadas de los tipos abstractos.

•Se permite que una clase abstracta contenga miembros abstractos, aunque no es necesario.

•No se puede sellar una clase abstracta.

\_

•Las clases abstractas se sitúan en la cima de la jerarquía de clases.

•Establecen la estructura y significado del código.

•Facilitan la creación de marcos de trabajo, esto es posible ya que las clases abstractas poseen una información y un comportamiento común a todas las clases derivadas de un marco de trabajo.

**(132) -** Las clases no-abstractas que derivan de una clase abstracta, ¿deben implementar todos sus métodos abstractos?

•Cuando una clase no abstracta se deriva de una clase abstracta, la clase no abstracta debe incluir implementaciones reales de todos los miembros abstractos heredados; por lo tanto, reemplaza a estos miembros abstractos.

**(133) -** Las clases abstractas que derivan de una clase abstracta, ¿deben implementar todos sus métodos abstractos?

No.

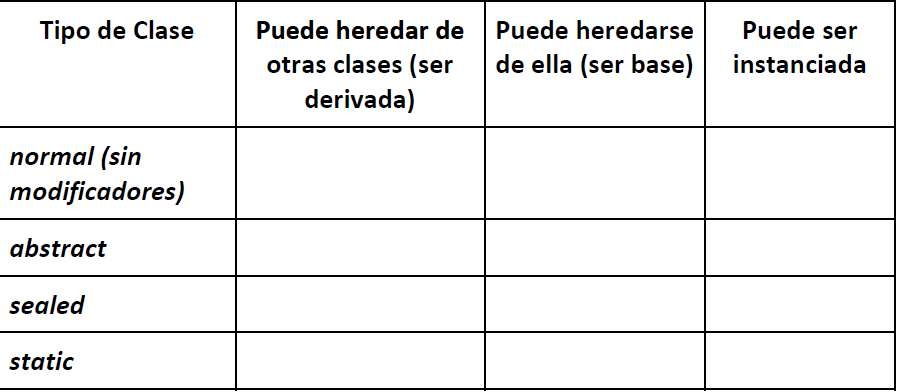
**(134) -** ¿Se pueden declarar **miembros abstractos** en clases no-abstractas?

No.

**(135) -** ¿Para sobrescribir un método se debe heredar de una clase abstracta?

no. Capas de alguna que tenga un virtual.

**(136) -** Llenar los campos de la siguiente tabla con SÍ o NO según corresponda.



Si si si

Si Si No

Si No Si

NO NO NO