Examen Final de Programación Orientada a Objetos 1
(19-12-25) Alumno:

1. ¿Cuál es la mejor descripción de un Objeto en Programación Orientada a Objetos (POO)?
__ A. Una plantilla o esquema para crear entidades.
__ B. Una instancia de una clase que posee estado (atributos) y comportamiento (métodos).
__ C. Un tipo de dato primitivo, como un entero o un booleano.
__ D. Una variable que almacena la referencia a una función.
2: ¿Qué representa una Clase en el contexto de la POO?
__ A. Un objeto ya inicializado con valores específicos.
__ B. La descripción o el plano que define la estructura y el comportamiento de sus objetos.
__ C. El proceso de herencia de propiedades de una clase a otra.
__ D. Un método especial utilizado para destruir un objeto.
3: ¿Cuál es el principio fundamental del Encapsulamiento?
__ A. Permitir que una clase herede propiedades de otra clase.
__ B. Ocultar los detalles internos de la implementación de un objeto y proteger su estado interno.
__ C. Permitir que un objeto tome múltiples formas.
__ D. Definir un contrato de métodos sin proporcionar su implementación.
4: ¿Qué modificador de acceso permite que un miembro sea visible solamente dentro de su propia clase?
__ A. Público (Public)
__ B. Protegido (Protected)
__ C. Privado (Private)
__ D. Paquete (Package-Private)
5: ¿Cuál es la principal ventaja de utilizar Herencia en el diseño de clases?
__ A. Garantiza que todos los objetos se creen en un solo hilo de ejecución.
__ B. Asegura que los datos de un objeto nunca sean modificados.
__ C. Promueve la reutilización de código al permitir que una clase hija adquiera propiedades y comportamientos de una clase padre.
__ D. Controla el flujo de ejecución de un programa mediante bloques 'try-catch'.
6: Dada una relación de Herencia, si la Clase A es la clase padre y la Clase B es la clase hija, ¿cuál de las siguientes afirmaciones es universalmente cierta?
__ A. La Clase B puede acceder directamente a los miembros privados de la Clase A.
__ B. Un objeto de la Clase B se puede tratar como si fuera un objeto de la Clase A.
__ C. La Clase A no puede definir métodos que la Clase B pueda sobreescribir.
__ D. La Clase A debe ser obligatoriamente una Clase Abstract A.
7: El concepto de Polimorfismo se refiere a:
__ A. La capacidad de un objeto de almacenar datos de diferentes tipos primitivos.
__ B. La reutilización de una clase existente para crear una clase nueva.
__ C. La capacidad de diferentes objetos de responder a un mismo mensaje (llamada a método) de maneras específicas a su propio tipo.
__ D. El uso exclusivo de propiedades privadas en todas las clases.
8: ¿Cuál es la principal diferencia conceptual entre la Sobrecarga (Overloading) y la Sobreescritura (Overriding) de métodos?
__ A. La Sobrecarga ocurre en la herencia; la Sobreescritura no.
__ B. La Sobrecarga cambia la signatura (parámetros); la Sobreescritura cambia la implementación manteniendo la misma signatura.
__ C. Ambas son formas de Polimorfismo en tiempo de ejecución.
__ D. La Sobrecarga aplica a clases abstractas; la Sobreescritura aplica a interfaces.
9: ¿Qué característica define a una Clase Abstracta?
__ A. Contiene solo constantes y métodos estáticos.
__ B. Es una clase que no puede ser instanciada directamente, pero puede contener métodos implementados y métodos abstractos.
__ C. Solo puede heredar de otra clase abstracta.
__ D. No puede tener constructores.
10: Si una clase hereda de una Clase Abstracta que tiene un método abstracto, ¿qué debe hacer la clase hija (concreta) obligatoriamente?
__ A. Declarar un nuevo método con el mismo nombre y parámetros en la clase hija.
__ B. Convertirse también en una clase abstracta, a menos que implemente el método abstracto.
__ C. Llamar al constructor de la clase abstracta antes de su propio constructor.
__ D. Hacer que el método abstracto sea privado en la clase hija.
11: Una Interfaz en POO se utiliza principalmente para:
__ A. Definir un tipo de dato que permite la herencia de propiedades y métodos con implementación completa.
__ B. Establecer un contrato de métodos que una clase debe implementar, enfocándose en la capacidad ('puede hacer') de la clase.
__ C. Almacenar un grupo de objetos de tipos de datos diferentes de manera eficiente.
__ D. Manejar errores no previstos en tiempo de compilación.
12: A diferencia de las Clases Abstractas, las Interfaces tradicionalmente tienen una restricción clave en su implementación. ¿Cuál es?
__ A. Una clase puede implementar múltiples Interfaces, pero solo puede heredar de una única Clase Abstracta/Concreta.
__ B. No pueden contener constructores.
__ C. Todos sus miembros son implícitamente públicos y abstractos.
__ D. Tradicionalmente, no pueden contener campos de instancia, solo constantes estáticas.
13: ¿Cuál es el propósito principal de las Colecciones en POO?
__ A. Definir un conjunto de métodos sin implementación.
__ B. Proporcionar estructuras de datos prefabricadas y optimizadas para almacenar y gestionar grupos de objetos.
__ C. Garantizar que un método sea ejecutado al menos una vez.
__ D. Permitir la creación de objetos a partir de un tipo de dato abstracto.
14: ¿Cuál es la característica fundamental que distingue a un Set (Conjunto) de una List (Lista) en colecciones POO?
__ A. Un Set permite elementos duplicados, mientras que una List no.
__ B. Un Set mantiene los elementos en un orden específico; una List no garantiza orden.
__ C. Una List garantiza el orden de los elementos mediante un índice, y un Set no permite elementos duplicados.
__ D. Una List almacena pares clave-valor; un Set almacena solo valores.
15: ¿Cuál es el paso fundamental y común que se debe realizar antes de leer o escribir en un Archivo de Texto?
__ A. Crear un objeto de la clase 'Excepcion'.
__ B. Convertir todo el contenido del archivo a un formato binario.
__ C. Abrir el archivo para establecer una conexión (stream) y preparar el acceso.
__ D. Definir una clase abstracta para manejar la lectura.
16: En el contexto de la manipulación de Archivos de Texto, ¿por qué es crítico cerrar el archivo una vez que se termina de usar?
__ A. Porque el contenido del archivo se borra automáticamente al no cerrarse.
__ B. Para liberar los recursos del sistema operativo asociados al archivo y asegurar que todos los datos en búfer se hayan escrito.
__ C. Para evitar que otros programas lean el archivo.
__ D. Para convertir los datos de texto a un formato comprimido.
17: ¿Qué busca lograr el Manejo de Excepciones en un programa POO?
__ A. Prevenir errores de sintaxis en tiempo de compilación.
__ B. Garantizar que todos los atributos de un objeto sean privados.
__ C. Proporcionar un mecanismo estructurado para gestionar errores o situaciones anómalas que ocurren durante la ejecución.
__ D. Implementar polimorfismo mediante la sobrecarga de métodos.
18: En el bloque de manejo de excepciones, ¿cuál es el propósito de la sección `finally` (o equivalente)?
__ A. Ejecutar código solo si ocurre una excepción.
__ B. Ejecutar código solo si no ocurre ninguna excepción.
__ C. Ejecutar código crucial de limpieza o liberación de recursos, independientemente de si ocurrió una excepción o no.
__ D. Definir las excepciones que pueden ser lanzadas por el método.
19: Desde la perspectiva de la POO, ¿qué significa que una Excepción sea un objeto?
__ A. Que todas las excepciones deben implementarse utilizando una interfaz.
__ B. Que una excepción no puede ser generada por el programador.
__ C. Que las excepciones tienen estado (información de error) y comportamiento (métodos, como el rastreo de pila), y pueden participar en la herencia.
__ D. Que las excepciones deben ser manejadas en el mismo bloque donde ocurren.
20: En el contexto de las Colecciones, ¿cuál es el beneficio clave de usar un tipo genérico (o parametrizado) como `List<T>` o `ArrayList<String>`?
__ A. Permitir el almacenamiento de elementos de cualquier tipo sin restricciones.
__ B. Forzar al programador a usar la palabra clave 'new' para crear la colección.
__ C. Proporcionar seguridad de tipos en tiempo de compilación y evitar la necesidad de conversiones (casts) manuales en tiempo de ejecución.
__ D. Hacer que la colección sea inmutable después de su creación.