POO-ENCAPSULAMIENTO-COMPOSICIÓN

Carrera Programador full-stack

EJERCICIOS

Conceptos básicos de POO:

- **Clase**: Molde para crear objetos con atributos y comportamientos
- **Atributo:** Propiedades que tiene un objeto (ej: nombre, velocidad)
- **Método:** Acción que puede hacer el objeto (ej: arrancar, tirar dado).
- **Instancia:** Un objeto creado a partir de una clase, cada instancia o cada objeto tiene un nombre único.
- **Encapsulamiento:** Mantener los datos protegidos dentro de la clase, mediante los modificadores de acceso: public y private. Los modificadores se colocan siempre delante de las propiedades y de los métodos
- **Composición:** Un objeto forma parte de las propiedades de otro objeto.

Aclaraciones:

- **Para cada ejercicio crear proyecto NPM individual y subirlos todos a un mismo repositorio de GitHub.**

- **No hay una sola forma de tener bien los ejercicios → lo que importa es saber justificar bien las decisiones que se tomen**

RECOMENDACIONES

- Priorizar siempre la legibilidad del código
  - Usar nombres descriptivos
- Si una clase tiene funcionalidades que no tienen nada que ver una con otra → separar clases
- Hacer siempre un planteo de lo que se va a implementar
- Pensar defensivamente: chequear siempre los parámetros que llegan
- Usar un archivo por clase → ‘nombreclase.ts’
- Evitar en la medida de lo posible el código duplicado
  - Métodos con código repetido, usar un método privado, y que ambas métodos lo invoquen

1- Plantear una clase Auto de la forma en que se vió en la clase → especificando variables internas y métodos. Implementar en TypeScript

2-Plantear una clase Persona, pensar que atributos y métodos puede tener e implementarla en Typescript.

3-Implementar la clase Televisor con todos los atributos y métodos, como se vio en clase, incluido la funcionalidad mute.

4-Crea una clase llamada Rectangulo que represente. Esta clase debe tener:

- Un constructor que reciba dos parámetros: ancho y altura.
- Métodos para calcular el área y el perímetro del rectángulo.
- El método calcular_area() debe retornar el área del rectángulo (ancho * altura).
- El método calcular_perimetro() debe retornar el perímetro del rectángulo (2 * (ancho + altura)).

Una vez definida la clase Rectangulo, crea una instancia de esta clase con un ancho de 5 unidades y una altura de 10 unidades. Luego, utiliza los métodos de la clase para calcular el área y el perímetro del rectángulo creado e imprime los resultados.

5-Implementar una clase llamada Libro que contenga los siguientes atributos: ISBN, Título, Autor, Número de páginas. La clase debe tener un método para cargar un libro pidiendo los datos al usuario y luego informar mediante otro método el número de ISBN, el título, el autor del libro y el número de páginas. Crear dos o tres instancias de la Clase e implementar ambos métodos.

6-Implementar una clase que simule el comportamiento de una Calculadora, con dos números y las cuatro operaciones básicas: suma resta, multiplicación y división. Todas la operaciones deben retornar un resultado, que luego va a ser mostrado por un método aparte. A su vez en las operaciones de multiplicación y división se debe validar que no se ingrese un valor igual a 0. Probar la calculadora con diferentes valores

7-Implementar una clase que simule el comportamiento de un dado de seis caras. La clase debe tener un atributo que almacena el valor actual del dado (un número entre 1 y 6). A su vez la clase cuenta con varios métodos.

- Método constructor que inicializa el dado con un valor aleatorio entre 1 y 6.
- Método que simula el lanzamiento del dado, asignando un nuevo valor aleatorio entre 1 y 6 al atributo valor del dado
- Método que devuelve el valor actual del dado.

Proba el funcionamiento del dado

8-Implementar a una clase que represente una cuenta bancaria, con dos atributos: Número de cuenta y Saldo actual. Implementa los siguientes métodos:

- Un constructor que me permita establecer el número de cuenta y el saldo.
- Un método que incremente el saldo.
- Un método que disminuya el saldo si hay fondos suficientes, caso contrario no hace nada.
- Un método que devuelva el saldo actual

9- Implementar una clase Decodificador y relacionarla con la clase Televisor del ejercicio 2.

10-Usando la Clase automotor del ejercicio 1, implementar la clase Registro Automotor, con métodos para:

- Agregar un auto
- Buscar un auto por patente
- Eliminar o actualizar autos existentes

11-Aplicar lo visto hasta esta clase para modelar un sistema educativo donde:

- Los profesores deben tener un listado de sus alumnos.
- Cada alumno debe saber su nota e informar si está aprobado o no (es decir si la nota es mayor que 7).
- La escuela debe tener un registro de los alumnos y maestros, y debe poder matricular/contratar y expulsar/despedir a los mismos.

12-Usando la clase Dado del ejercicio 6, simular el comportamiento de un cubilete de 5 dados. Implementar los métodos necesarios para que se puedan lanzar todos los dados o algún dado o dados en particular.

13- Usando la clase libro del ejercicio 5, simular el funcionamiento de un sistema para gestionar una colección de libros, permitiendo realizar las operaciones básicas: alta, baja, modificación y consulta.

La clase Gestor de Libros debe permitir realizar las siguientes acciones:

- Insertar un libro: agregar un nuevo libro a la colección.
- Consultar un libro: buscar un libro y mostrar sus datos.
- Modificar un libro: cambiar los datos de un libro ya existente (por ejemplo: autor, año, etc.).
- Eliminar un libro: realizar una baja lógica, es decir, el libro no se borra definitivamente, sino que se marca como inactivo o dado de baja.

Consideraciones importantes:

- Antes de realizar cualquier operación (incluso insertar), se debe verificar si el libro ya existe en la colección.
- El estado de cada libro (activo o dado de baja) debe formar parte de la clase Libro,

14- Usando la Clase persona del Ejercicio 2 y la clase Cubilete del ejercicio anterior, implementar un juego de Generala. Esta Generala solo va a considerar puntos por obtener la siguientes marcas:

- Escalera : 20 puntos
- Full: 30 puntos
- Poker: 40 puntos
- Generala: 50 puntos

En cada juego, cada jugador tiene 3 tiros y gana el jugador que tenga más puntos. En caso de que los 3 tiros ningún jugador haya hecho una marca, se declara empate. Implementar un método de desempate a su elección.

En caso de ser necesario, modificar la clase persona para agregarle más atributos.