Parcial #2: Clases, arreglos y funciones predefinidas

Objetivo

Desarrollar un sistema dinámico de gestión de biblioteca utilizando PHP, aplicando conceptos avanzados de programación orientada a objetos, incluyendo herencia de clases, interfaces, manejo de arreglos, uso de funciones predefinidas, procesamiento de parámetros GET, ordenamiento y filtrado de datos.

Instrucciones

1. Creen una interfaz llamada Prestable con el siguiente método: obtenerDetallesPrestamo(): string
2. Modifiquen la clase RecursoBiblioteca para que implemente la interfaz Prestable. Cambiar el modifier a abstract
3. Creen las siguientes clases que hereden de RecursoBiblioteca:
    - Libro: Agregar un atributo isbn (cadena)
    - Revista: Agregar un atributo numeroEdicion (entero)
    - DVD: Agregar un atributo duracion (entero, en minutos)
4. Implementen el método obtenerDetallesPrestamo() en cada clase hija, proporcionando una implementación específica que devuelva una cadena con los detalles particulares de cada tipo de recurso.
5. Creen un arreglo asociativo llamado $estadosLegibles que mapee los valores de estado a sus versiones legibles:
    - 'disponible' debe mostrarse como 'DISPONIBLE'
    - 'prestado' debe mostrarse como 'PRESTADO'
    - 'en_reparacion' debe mostrarse como 'EN REPARACIÓN'
6. Modifiquen la clase GestorBiblioteca para que utilice las nuevas clases hijas al cargar los recursos desde el JSON.
7. Implementen los siguientes métodos en la clase GestorBiblioteca:
    - agregarRecurso(RecursoBiblioteca $recurso)
    - eliminarRecurso($id)
    - actualizarRecurso(RecursoBiblioteca $recurso)
    - actualizarEstadoRecurso($id, $nuevoEstado)

- buscarRecursosPorEstado($estado)
- listarRecursos($filtroEstado = '', $campoOrden = 'id', $direccionOrden = 'ASC')

8. Implementen la lógica en el archivo index.php para manejar las diferentes acciones (agregar, editar, eliminar, cambiar estado, listar, ordenar y filtrar) utilizando los métodos de GestorBiblioteca.
9. Modifiquen la tabla en index.php para incluir una columna que muestre los detalles de préstamo utilizando el método obtenerDetallesPrestamo().
10. Asegúrense de que el formulario para agregar/editar recursos maneje correctamente los campos específicos de cada tipo de recurso (ISBN para libros, número de edición para revistas, duración para DVDs).
11. Implementen un manejo básico de errores para parámetros inválidos o acciones no reconocidas.
12. Asegúrense de que todas las operaciones (agregar, editar, eliminar, cambiar estado) actualicen correctamente el archivo JSON.

Restricciones

1. Todo el manejo de datos persistentes debe hacerse a través del archivo JSON.
2. El código debe estar bien comentado y seguir las mejores prácticas de programación orientada a objetos.
3. Utilicen únicamente GET para el envío de datos.

Entregables

1. Un archivo PHP que contenga todas las definiciones de clases e interfaces (puede ser nombrado como clases.php).
2. El archivo principal index.php que contenga la lógica de procesamiento de acciones y la interfaz de usuario.
3. El archivo JSON biblioteca.json que se utilizará para almacenar los datos de los recursos.

Criterios de Evaluación

1. Correcta implementación de la interfaz y las clases abstractas y concretas (25%)
2. Funcionalidad completa del gestor de biblioteca (agregar, editar, eliminar, cambiar estado, listar) (30%)

3. Implementación correcta del filtrado de recursos (20%)
4. Correcta persistencia y recuperación de datos usando JSON (15%)
5. Manejo adecuado de errores y validación de entrada (5%)
6. Código limpio, bien comentado y que siga las mejores prácticas de POO (5%)

Tiempo máximo de desarrollo

2 horas

Entrega:

- Ambos integrantes del grupo deben tener el código completo en su carpeta PARCIAL_2 dentro de PARCIALES en su repositorio personal.
- En la entrega de Microsoft Teams, cada integrante debe subir un archivo de texto con los nombres de ambos miembros del grupo.

Notas adicionales

1. Se proporciona un código base en index.php y clases.php que incluye la estructura básica de la aplicación. Los estudiantes deben completar la implementación de las clases, métodos y lógica de negocio.
2. El archivo biblioteca.json contiene datos de ejemplo para comenzar. Los estudiantes deben asegurarse de que la aplicación pueda leer y escribir correctamente en este archivo.
3. La interfaz de usuario ya está implementada en index.php. Los estudiantes deben enfocarse en la lógica backend y en la integración con la UI existente.