PROGRAMACIÓN ORIENTADA A OBJETOS
Laboratorio N°3

Ingeniería Civil en Informática - Departamento de Ciencias de la Ingeniería

Jueves 27 de Noviembre de 2025

UNIVERSIDAD DE LOS LAGOS

Objetivo de la Evaluación
Aplicar los principios de Herencia Múltiple, Polimorfismo y el Principio de Sustitución de Liskov (LSP) para diseñar una arquitectura de software robusta.

Reglas de Integridad Académica
- Trabajo individual. Cualquier intento de copia o respuestas idénticas entre estudiantes se calificará con nota mínima.
- No se permite el uso de dispositivos electrónicos como smartphones,
- Se pueden utilizar solo apuntes de su repositorio personal de GitHub.
- El uso de IA Generativa para resolver el problema será sancionado con la nota mínima.

Contexto del Problema
Usted es el arquitecto de software principal de **StreamPy**, una nueva plataforma de streaming. Necesita diseñar la jerarquía de clases que manejará los diferentes tipos de contenido. No todo el contenido se comporta igual, y las funcionalidades (como reproducir o calificar) deben estar correctamente distribuidas para evitar violaciones a los principios SOLID.

Requisitos Funcionales y Conceptos a Aplicar
La plataforma debe manejar los siguientes conceptos:

Concepto	Requisito POO	Descripción
LSP	Diseño seguro	Asegurar que cualquier función que espere un tipo de contenido reproducible (**Reproducible**) funcione correctamente con cualquiera de sus subtipos sin lanzar errores.
Herencia Múltiple	Combinar capacidades	Un tipo de contenido específico (**Miniserie**) debe heredar capacidades funcionales de múltiples fuentes no relacionadas (por ejemplo: *ser Reproducible y ser Calificable*).
Polimorfismo	Comportamiento dinámico	Un método genérico como **ejecutar_accion()** debe invocar comportamientos específicos (como **reproducir()** o **mostrar_detalle()**) que varían según el tipo de objeto real (una **Película** se reproduce diferente a un **Documental**).

Estructura de Clases Requerida

A. La Clase Base Principal

1. **Contenido:**
   - **Atributos:** *titulo(str)*, *anio(int)*.
   - **Método:** **mostrar_detalle()**: Imprime el título y el año.

B. Estructuras de Clases

Definir las siguientes clases para dotar de capacidades a otros contenidos.

2. **Reproducible(Clase Normal):** Definir la interfaz de reproducción.
   - **Método:** **reproducir()**: Debe ser implementado directamente por las subclases que hereden de ella (o se puede dejar vacío si se desea que sea solo un marcador de herencia, aunque se recomienda la implementación para forzar el comportamiento).
3. **Calificable (Clase de Capacidad):** Añade la funcionalidad de calificación.
   - **Atributo:** *rating* (float, inicializado en 0.0).
   - **Método:** **calificar(puntuación)**: Actualiza el rating e imprime la nueva calificación.

C. Subclases

Estas clases deben usar la herencia simple o múltiple según sea necesario.

4. **Película:**
   - Hereda de **Contenido** y **Reproducible**.
   - **Atributo Extra:** *duracion_minutos(int)*.
   - **Polimorfismo:** Implementa reproducir().
5. **Documental:**
   - Hereda de Contenido y No de Reproducible.
   - **Atributo Extra:** *tema(str)*.
   - **LSP: Justificación:** Como Documental no hereda de la Interfaz Reproducible, la función **lista_de_reproduccion** debe poder manejarlo de forma segura (Duck Typing).
6. **Miniserie (Punto de Herencia Múltiple):**
   - Hereda de Contenido, Reproducible y Calificable.
   - **Atributo Extra:** *num_episodios(int)*.
   - **Polimorfismo:** Implementa **reproducir()**.

Verificación de Principios

Una vez definidas todas las clases, se deben las siguientes funciones para probar su diseño:

A. Prueba de LSP y Polimorfismo

Crear la función **lista_de_reproduccion(lista_contenidos)**: Recibe una lista mixta de objetos, pero solo llama a **reproducir()** si el objeto cumple la Interfaz Reproducible (mediante Duck Typing). Crear una función que aplique el **Polimorfismo Dinámico** y asegure el **LSP**.

B. Prueba de Herencia Múltiple y Polimorfismo

Instanciar un objeto de cada clase, especialmente de **Miniserie**, y demostrar que se puede ejecutar todos los métodos heredados de sus múltiples padres (**reproducir()**, **calificar()**, y **mostrar_detalle()**).

Entregables

1. Crear un archivo Python (*streaming_poo.py*) con la implementación completa de todas las clases y métodos descritos incluyendo el uso del módulo Reproducible.
2. Incluir la función de prueba **lista_de_reproduccion(lista_contenidos)** y un bloque de ejecución que demuestre la prueba de **Polimorfismo y LSP**.
3. Demostrar el uso de la **Herencia Múltiple** en la clase **Miniserie** al llamar a sus tres métodos funcionales.