ACTIVIDAD Y TIPO: "Interfaces"/ AEF
LUGAR DE REALIZACIÓN: Centro docente/Casa

Unidad Didáctica	Criterios de Evaluación	Resultados de Aprendizaje
UD12	RA4.i) Se han definido y utilizado interfaces.	RA4
UD11/UD12	RA7.b) Se han utilizado modificadores para bloquear y forzar la herencia	RA7
UD11/UD12	RA7.d) Se han creado clases heredadas que sobrescriban la implementación de métodos de la superclase.	RA7
UD11/UD12	RA7.e) Se han diseñado y aplicado jerarquías de clases.	RA7
UD11/UD12	RA7.f) Se han probado y depurado las jerarquías de clases.	RA7
UD11/UD12	RA7.g) Se han realizado programas que implementen y utilicen jerarquías de clases.	RA7
UD11/UD12	RA7.h) Se ha comentado y documentado el código.	RA7

Objetivos Generales	Competencias Profesionales, Personales y Sociales
a) e) f) h) i) j) n) ñ) q) r) w)	a) e) g) h) i) j) n) ñ) q) r) w)

- **Individual/ Equipo:** Individual
- **Descripción de la actividad:** Realizar la actividad propuesta.
- **Fecha y medio de entrega:** La fecha y medio de entrega se indicará en la publicación de la tarea en Classroom. Si se incumplen las especificaciones indicadas en la actividad tanto de contenido como de fecha, formato, nombre o forma de entrega, la actividad será calificada con un 0.
- **Formato:** Se deberá entregar un archivo **Ape1Ape2Nom.zip** conteniendo:
    - un proyecto Netbeans de nombre **Ape1Ape2Nom** conteniendo la implementación de la actividad con la especificación indicada en la misma.
    - **Ud12Act1.png** con el UML de la actividad 4.

INSTRUMENTOS DE EVALUACIÓN:
La calificación de la competencia INICIATIVA supondrá el 7.5% de la calificación de cada CE.

CE	**INSTRUMENTO DE EVALUACIÓN**	**% CE Actividad**
RA4.i) Se han definido y utilizado interfaces.	Checklist	60% del 10%
RA7.b) Se han utilizado modificadores para bloquear y forzar la herencia	Checklist	40% del 10%
RA7.d) Se han creado clases heredadas que sobrescriban la implementación de métodos de la superclase.	Checklist	40% del 15%
RA7.e) Se han diseñado y aplicado jerarquías de clases.	Checklist	40% del 15%
RA7.f) Se han probado y depurado las jerarquías de clases.	Checklist	40% del 15%
RA7.g) Se han realizado programas que implementen y utilicen jerarquías de clases.	Checklist	40% del 20%
RA7.h) Se ha comentado y documentado el código.	Checklist	40% del 5%

Ud12Act1. Crearemos una clase llamada **Serie** con las siguientes características:

- Sus atributos son **titulo, numero de temporadas**, **entregado, genero y creador.**
- Por defecto, el número de temporadas es de 3 temporadas y entregado **false**. El resto de atributos serán valores por defecto según el tipo del atributo.
- Los constructores que se implementarán serán:
  - Un constructor por defecto.
  - Un constructor con el título y creador. El resto por defecto.
  - Un constructor con todos los atributos, excepto de entregado.

- Los métodos que se implementara serán:
  - Métodos get de todos los atributos, excepto de entregado.
  - Métodos set de todos los atributos, excepto de entregado.
  - Sobrescribe los métodos toString.

Crearemos una clase **Videojuego** con las siguientes características:

- Sus atributos son **titulo, horas estimadas, entregado, genero y compañia**.
- Por defecto, las horas estimadas serán de 10 horas y entregado false. El resto de atributos serán valores por defecto según el tipo del atributo.
- Los constructores que se implementarán serán:
  - Un constructor por defecto.
  - Un constructor con el titulo y horas estimadas. El resto por defecto.
  - Un constructor con todos los atributos, excepto de entregado.

- Los métodos que se implementarán serán:
  - Métodos get de todos los atributos, excepto de entregado.
  - Métodos set de todos los atributos, excepto de entregado.
  - Sobrescribe los métodos toString.

Como vemos, en principio, las clases anteriores no son padre-hija, pero sí tienen bastante en común, por eso vamos a hacer una interfaz llamada **Entregable** con los siguientes métodos:

- **entregar()**: cambia el atributo prestado a true.
- **devolver()**: cambia el atributo prestado a false.
- **isEntregado()**: devuelve el estado del atributo prestado.
- Método **compareTo (Object a)**, compara las horas estimadas en los videojuegos y en las series el número de temporadas. Como parámetro que tenga un objeto, no es necesario que implementes la interfaz Comparable. Recuerda el uso de los casting de objetos.

Implementa los anteriores métodos en las clases Videojuego y Serie. Ahora crea una aplicación ejecutable de nombre Prueba y realiza lo siguiente:

- Crea dos estructuras, uno de **Series** y otro de **Videojuegos**, de 5 posiciones cada uno.
- Carga las estructuras usando distintos constructores.
- Entrega algunos **Videojuegos** y **Series** con el método **entregar**().
- Cuenta cuántas **Series** y **Videojuegos** hay entregados. Al contarlos, muéstralos.
- Por último, indica el **Videojuego** que tiene más horas estimadas y la serie con más temporadas. Muéstralos en pantalla con toda su información (usa el método toString()).