Examen UD02 - Ficheros

Un amigo cinéfilo quiere montar un festival de proyecciones de películas galardonadas con premios Oscar. Como buenos amigos, vamos a ayudarle a gestionar la información de dichas películas para que pueda filtrar información que le sea útil para las posibles dudas que tengan los asistentes del festival.

Para ello, partiremos de 2 ficheros de texto, los cuales nos proporciona:

- oscar_age_female.csv
- oscar_age_male.csv

Como podéis ver, estos ficheros contienen información los ganadores al oscar a mejor actor y mejor actriz, la edad a la que consiguieron el galardón, el nombre de la película y el año en que consiguieron el galardón.

Para poder gestionar toda esta información, vamos a trabajar con 3 clases de partida:

(**IMPORTANTE**, los nombres de los atributos no se pueden modificar, es decir, el atributo “nombre” de la clase Actor debe tener ese nombre. Su modificación supondrá pérdida de nota en este ejercicio):

- La clase **PeliculaOscarizada**, que contenga la siguiente información leída de los ficheros CSV:
    - pelicula: String con el nombre de la película
    - anyo: Int con el año en que la película ganó el Oscar a mejor actor o actriz
    - actor: String con el nombre del actor/actriz
    - edad: Int con la edad del actor/actriz en el momento de conseguir el Oscar
    - sexo: String que podrá ser “H” o “M”
- La clase **Actor**, que utilizarás para escribir sobre un fichero en formato JSON, en la cual deberás almacenar los siguientes atributos.
    - nombre: String con el nombre del actor/actriz
    - sexo: String que podrá ser “H” o “M”
    - anyoNacimiento: con el año de nacimiento del actor/actriz
    - peliculas: lista de objetos “Pelicula” por las que el actor/actriz ha ganado el Oscar
- La clase **Pelicula**, con los siguientes atributos:
    - titulo: String con el título de la película
    - anyo: Int con el año en que la película ganó el Oscar

Además, deberás crearte una clase llamada **Utilidades**, en la cual crearás todos los métodos necesarios para manipular la información que necesites de las clases anteriores. Dichos métodos serán los siguientes:

- **leerPeliculasOscarizadasCsv**: lee un fichero CSV y devuelve una lista de objetos PeliculaOscarizada. Debe tener en cuenta también el parámetro sexo para filtrar por sexo.
- **convertirPeliculasOscarizadasEnActores**: dada una lista de PeliculasOscarizadas, deberás devolver una lista de objetos Actor, en la que estarán incluidos todos los actores y actrices.
- **escribirActoresenJson**: dado una lista de objetos Actor, escribe en un fichero JSON la lista de actores/actrices en el formato solicitado.

Opcionalmente, crearás también dos métodos más:

- **actoresConMasdeUnOscar**: devuelve una lista de Strings con el actor y la actriz más jóvenes en ganar un Oscar.
- **actoresMasJovenesEnGanarUnOscar**: dada una lista de PeliculasOscarizadas, deberás devolver una lista de nombres de actores o actrices que hayan ganado un oscar, ordenado de forma ascendente.

Ejercicio 1

Almacenar en una lista de objetos PeliculasOscarizadas la información obtenida de los 2 ficheros .csv.

Ejercicio 2

Dada la información de los ficheros .csv facilitados, crea un fichero llamado **actores.json**, en el que se almacenarán todos los actores y actrices que hayan ganado algún oscar, así como las películas con las que lo han ganado.

Este fichero actores.json deberá guardarse en un directorio llamado salida, el cual deberás crear dentro de resources. El fichero actores.json deberá tener un aspecto parecido al siguiente:

```
[ {
 "name" : "Sidney Poitier",
 "sex" : "H",
 "yearOfBirth" : 1926,
 "movies" : [ {
   "title" : "Lilies of the Field",
   "year" : 1963
 } ]
}, {
 "name" : "Spencer Tracy",
 "sex" : "H",
 "yearOfBirth" : 1900,
 "movies" : [ {
   "title" : "Captains Courageous",
   "year" : 1937
} ]
}, ...
]
```

Ejercicio 3

Crea la clase UtilidadesTest, donde compruebes el funcionamiento del método actoresConMasDeUnOscar, que devuelve una lista con los nombres de los actores y actrices con más de un Oscar.

Ejercicio Extra

- Crea el método actoresConMasdeUnOscar testado anteriormente en la clase Utilidades.
- Crea el método actoresMasJovenesEnGanarUnOscar, que devuelve una lista de Strings con el actor y la actriz más jóvenes en ganar un Oscar.

Rúbrica de calificación.

- Cada Criterio de Evaluación (CE) se valorará en 1.66 ptos.
- El ejercicio extra supondrá 1 punto extra si el examen está superado.

CE1a. Se han utilizado clases para la gestión de ficheros y directorios.			
Excelente (10)	Bueno (7,5)	Aceptable (5)	Insuficiente (0)
Resuelve la gestión de ficheros y directorios usando las **clases Path, Paths y Files**.	Resuelve la gestión de ficheros y directorios usando la **clase File**.	Un problema no se ha resuelto adecuadamente: existencia, rutas relativas al proyecto, etc.	Dos o más problemas no se han resuelto adecuadamente.
CE1c. Se han utilizado clases para recuperar información almacenada en ficheros.			
Excelente (10)	Bueno (7,5)	Aceptable (5)	Insuficiente (0)
Utiliza **métodos** de la **clase Files** para la lectura del **fichero CSV**.	Utiliza **métodos** de **otras clases** para la lectura del **fichero CSV**.	El **parsing** del fichero CSV al objeto no es adecuado, pero es **funcional**.	El **parsing** del fichero CSV no **funciona**.
CE1d. Se han utilizado clases para almacenar información en ficheros.			
Excelente (10)	Bueno (7,5)	Aceptable (5)	Insuficiente (0)
Utiliza **métodos** de la **librería Jackson** para la escritura del **fichero JSON**.	Utiliza **métodos** de **otras librerías** para la escritura del **fichero JSON**.	El **fichero JSON** escrito no sigue la sintaxis solicitada, pero **es funcional**.	El fichero JSON no se escribe o no es funcional.
CE1e. Se han utilizado clases para realizar conversiones entre diferentes formatos de ficheros.			
Excelente (10)	Bueno (7,5)	Aceptable (5)	Insuficiente (0)
Convierte eficientemente el objeto leído en CSV al objeto a escribir en JSON.	Requiere de otras clases adicionales para la conversión entre objetos.	Convierte los objetos, pero modificando atributos.	No se realiza ningún tipo de conversión.
CE1f. Se han previsto y gestionado las excepciones.			
Excelente (10)	Bueno (7,5)	Aceptable (5)	Insuficiente (0)
Se capturan excepciones con **try-with-resources** y se crean **excepciones propias**.	Captura excepciones con **try-with-resources** o se crean excepciones propias.	Se capturan excepciones, pero sin try-with-resources y no se crean propias.	No se capturan ni gestionan excepciones.
CE1g. Se han probado y documentado las aplicaciones desarrolladas			
Excelente (10)	Bueno (7,5)	Aceptable (5)	Insuficiente (0)
Realiza y prueba un método de prueba usando **Junit5**.	Realiza un **esqueleto** de prueba de funcionalidad usando **Junit5**.	Realiza un **esqueleto** de prueba de funcionalidad, pero **no parece adecuada**	No realiza código para pruebas.