Grado en Ingeniería Informática Plan 40
Examen de Programación I
Convocatoria ordinaria (26/01/2021)

Enunciado

Se desea construir una aplicación con interfaz gráfica en Java para la gestión de un registro de juegos. Los juegos se caracterizan por tener un título, un desarrollador, una plataforma en la que se distribuye y la edad recomendada de los usuarios del juego. Dos juegos se consideran iguales si tienen el mismo título y son de la misma plataforma.

En esta aplicación el usuario tendrá en una ventana principal (Figura 1) un listado de los juegos registrados; desde esta ventana se podrá: introducir nuevos juegos, cambiar el orden en que se muestra el listado y pedir una estadística de número de juegos en cada plataforma.

En el diseño de la aplicación se han identificado dos clases para representar la información: **Game** que servirá para representar cada uno de los juegos y **RegisteredGames** que se usará para representar el conjunto de juegos registrados.

En la interfaz gráfica se han identificado inicialmente una ventana principal `MainWindow` (Figura 1) y un diálogo `NewGame` (Figura 2).

El diseño de la aplicación que se plantea, requiere la realización de las clases **Game**, y **RegisteredGames**, en el paquete **games**.

La clase **Game** se suministra desarrollada **casi por completo**. Vaya al código que se suministra y revise los métodos disponibles. El método `toString` de un objeto de la clase **Game** devuelve una String formada por la concatenación del título, el desarrollador, la plataforma y la edad separados entre ellos por el carácter espacio seguido de una barra vertical y de otro espacio, `(" | ")`.

La clase **RegisteredGames** no está desarrollada y deberá tener los siguientes métodos disponibles:

Método	Descripción
boolean add(Game)	Añade un nuevo juego no repetido y devuelve verdadero. Si ya existe, no lo añade y devuelve falso.
boolean remove(String, String)	Elimina el juego que tenga el título y la plataforma pasados por parámetros. Devuelve verdadero si lo elimina y falso si no se encuentra.
void setOrder(**int** ord)	Establece el orden en que se genera el listado de juegos con getList ord=0: Por título y plataforma ord=1: Por edad, título y plataforma
List<Game> getList()	Devuelve la lista de juegos en el orden establecido por setOrder
Set<String> getListPlatforms()	Devuelve un conjunto de ristras en orden alfabético. Cada ristra es la concatenación del nombre de la plataforma seguido del carácter dos puntos (":") y seguido del número de juegos para dicha plataforma.
String toString()	Devuelve la lista de juegos en el orden establecido y separados por saltos de línea. La información que se muestra de cada juego es su toString.

Parte 1 (50%)

Realice el desarrollo de las distintas clases según los requisitos descritos en el enunciado anterior. En este proceso puede necesitar añadir atributos, redefinir métodos de la clase **Object**, implementar comparadores, etc. Use los contenedores de la librería de Java más adecuados en cada caso. Las modificaciones que se hagan

en los contenedores devueltos por los métodos en que esté especificado, no tienen que afectar a los contenedores de los objetos que requieran un contenedor.

Parte 2 (25%)

Desarrolle la clase llamada **RegisteredGamesTest** en la que debe probar adecuadamente la funcionalidad de la clase **RegisteredGames**, usando JUnit. Dicha clase debe pertenecer al paquete: **games.test**.

Para facilitar las pruebas se ha implementado en la clase `RegisteredGames` el método `fill()` que añade juegos prefijados al propio objeto. Para ejecutar la clase de pruebas es necesario incluir, en el método `main` de la clase `RegisteredGamesTest` la instrucción:

```
org.junit.runner.JUnitCore.main("games.test.RegisteredGamesTest");
```

Parte 3 (25%)

La clase diálogo **NewGame** se suministra completamente desarrollada (ver el código suministrado para conocer los detalles).

La clase **MainWindow** representa la ventana principal (Figura 1) y se encarga de mostrar en un `JTextArea` (no modificable por el usuario) el listado de juegos registrados en todo momento. Debe suministrar dos botones: uno para añadir un nuevo juego y otro para generar otro diálogo en la clase **PlataformsReport** en el que se muestre el listado de plataformas con el número de juegos en cada una (se obtiene con el método `getListPlatforms` y puede usarse un `JTextArea` para mostrarlo). También debe permitir seleccionar entre los tipos de ordenación. Al seleccionar un orden distinto debe actualizar apropiadamente el listado de juegos. Las ventanas se deben parecer lo más posible a las de las figuras.

Figura 1

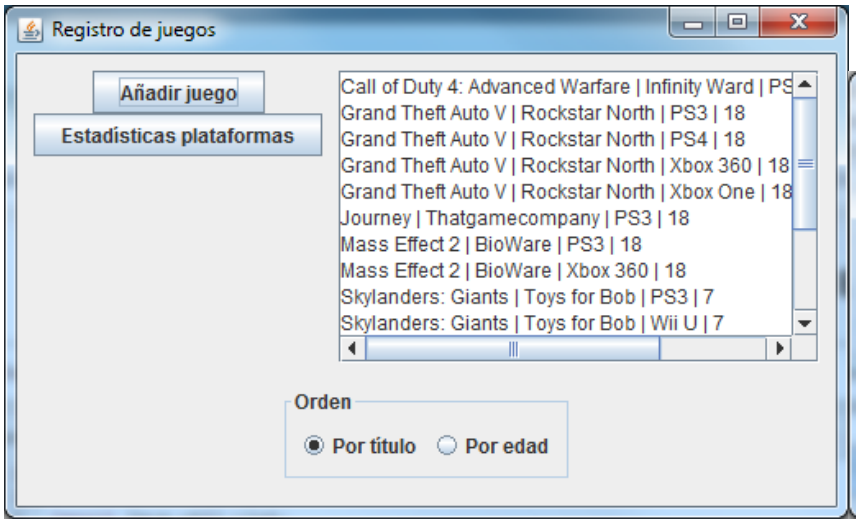

Figura 2

Para conseguir el borde alrededor de los tipos de ordenación con el texto "Orden" use `TitledBorder`. En el diálogo se usa un `JList` inicializado con un array de Strings {“3+”, “7”, “12+”, “16+”, “18+”} y tomando la opción seleccionada de su método `getSelectedIndex`.