# Ejercicio Obligatorio. Sesión 4

## 1. Instrucciones

Este ejerciciom deberá ser entregado **48 horas antes** de la siguiente clase de laboratorio.

## 2. Ejercicio 1. Ampliación del proyecto Dome

Tomando como punto de partida el proyecto Dome implementado en la sesión 4:

### 2.1 Añade un precio a cada item

1. Añade un precio base (`basePrice`, de tipo `double`) a cada item. Haz que se reciba como último parámetro con el resto de los datos, en el constructor. Asegúrate de que el precio no supera los 10000€ y no es negativo.
2. Además, cada ítem debe ser capaz de **calcular** su precio final, como sigue:
    - En caso de los Cds, se les añade un impuesto `TAX` de 2€.
    - Los dvds tienen el precio base como precio final
    - Los videoGames tienen un precio incrementado del 10% del precio base.
3. Añade a la librería multimedia un método `totalValue()` que devuelva un valor de tipo `double` con la suma de los precios finales de todos los ítems que tenga la librería.
4. Realiza las pruebas unitarias para todos los nuevos métodos creados.

### 2.2 Genera un código para los items

Añade un método a la librería Multimedia llamado `generateCode()` que devuelve una cadena con el código de todos los ítems de la base de datos separados por un guion. El código será una cadena formada por las tres primeras letras del título y un número secuencial, comenzando en 0. Por ejemplo si el título del primer elemento es "Yesterday, y el del segundo es "All you need is Love", el código que vuelca es "Yes0-All1". Si no hay ítems, devuelve "".

Pista: busca información sobre el método substring de la clase String en la documentación de Oracle (Oracle help centre).

Añade código a la aplicación (LibraryApp) para que imprima también el código generado.

```
------------------Código generado por los Items:
The0-Alc1-Gam2
```

Comprueba la funcionalidad con JUnit.

### 2.3 Search y equals

1. Cambia la implementación del método `search(...)` en la clase `MediaLibrary` sustituyendo el método `isLike (o isEqualTo)` por `equals` para realizar comparaciones entre diferentes tipos de `Item`.
    - Desarrolla las pruebas JUnit para verificar el método `search(...)`

### 2.4 Uso de toString

- Reemplaza el método `print` en las clases del proyecto `DOME` por el método `toString`.

2. Escribe las pruebas JUnit para verificar el método `toString()` en todas las subclases de `Item`.

Finalmente, si por alguna razón no has hecho aún alguna actividad obligatoria de tareas anteriores que afecten al proyecto DOME (añadir VideoGame, etc.) complétalo y envíalo. Será utilizado en la próxima sesión.

## 3. Ejercicio 2. Ampliación del proyecto red social

Importa el proyecto post de la tarea de las sesión 3 y cambia el nombre del proyecto y de sus paquetes para reflejar la nueva sesión de laboratorio:

Nombre proyecto: apellido1_apellido2_nombre_session4_task_post
Nombre paquete: uo.mp.s4.post.model y otros

3. Crea el método `toString()` en las diferentes clases de posts y modifica el método `print` para que use el `toString()` para imprimir en el parámetro `out`.
4. Añade un método a la red denominado toH`tmlFormat()` que permite formatear los posts. Este método devuelve una lista de cadenas con todos los post formateados en html, teniendo en cuenta que:
   a. En caso de que el post sea un mensaje el formato html que devuelve será:

       mensaje .

   b. En caso de que sea una foto devolverá:

      <img src= ”nombrefichero”>título</img>

5. Añade código al método principal de arranque, para que muestre por pantalla toda la lista que se genera con los post en formato html, de forma que salga uno en cada línea.

Ejemplo:

```
Post en formato HTML
<img src = foto1> </img>
<p> Me voy al cine</p>
<img src = foto2> </img>
<p> Estoy en Africa</p>
```