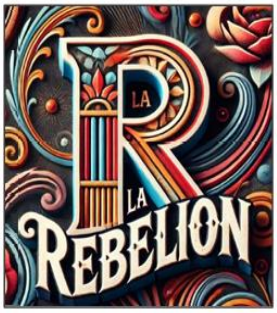
Práctica 1. "*El Formiguero*" y "*La Rebelión*"



➔ INTRODUCCIÓN

El mundo de la televisión últimamente parece estar en guerra. Algunos "se rebelan" contra otros, en directo, los otros desmienten las informaciones... Y los telespectadores, lejos de analizarlo de forma crítica, se hacen palomitas para ver según qué programa en función de su ideología.

➔ PROBLEMA A RESOLVER

Jordi está harto de la división entre "las dos Españas", y decide intentar poner un poco de paz creando una aplicación que recoja datos diarios sobre estos programas para dar objetividad a las distintas informaciones que van apareciendo.

Paso 1. Construcción de las clases que van a interactuar entre ellas

La aplicación que necesita desarrollar nuestro protagonista constará de los siguientes elementos:

- Una clase que contendrá al programa principal llamado "***AppProgramas***".
- Una clase ***Cadena***, con los siguientes atributos:
    - *nombre* (*String*)
    - *listaProgramas* (*ArrayList<Programa>)*

  Por defecto, se crea sin programas y se van agregando conforme se van *creando* programas relacionados con nuestra cadena.

- Una clase ***Programa***, con los siguientes atributos:
    - *nombre (String)*
    - *cadena (Cadena)*
    - *temporadas (int)*

- *listaEmpleados (ArrayList<Empleado>)*
- *listaInvitados (ArrayList<Invitado>)*
- *director (Empleado) – créalo en el mismo constructor y añádelo a la listaEmpleados*

Por defecto, se crean con 0 temporadas y no tendremos empleados (excepto el director) ni invitados hasta que se vayan añadiendo conforme se va *contratando* o *invitando*.

- Una clase ***Empleado***, con los siguientes atributos:
  - *id (String autogenerado EP001, EP002,…EP014,…).*
  - *nombre (String)*
  - *cargo (String)* – valores posibles: ***"director", "técnico", "presentador", "colaborador"***. Cualquiera que no sea uno de esos valores, **no se debe permitir añadirlo** como cargo y lo dejaremos con el valor por defecto **"*pte*"**.
  - *director (Empleado)* – debe coincidir con el *director* del *Programa*. **Si el *cargo* ya es "*director*", este atributo debe ser *nulo***.
- Una clase ***Invitado***, con los siguientes atributos:
  - *nombre (String)*
  - *profesión (String)*
  - *fecha_visita (LocalDate)* – por defecto *el día que se crea el Invitado*
  - *temporada (int)*

📢 A tener en cuenta…

- Las clases de tipo *Empleado* e *Invitado* se relacionan mediante composición con la clase *Programa*, de forma que si un objeto de tipo *Programa* se destruye, desaparecen también sus empleados e invitados asociados.
- La clase *Cadena* y la clase *Programa* tienen relación de tipo asociación/agregación bidireccional, es decir, se conocen la una a la otra y ambas existen por sí solas. De forma que, si el día de mañana el programa "*La rebelión*" vuelve a su *Cadena* original, esta sigue manteniendo todas sus características y solamente necesitaremos modificar el valor del atributo *cadena* en la clase *Programa* y eliminar de la *listaProgramas* en la clase *Cadena* dicho programa.
- Ten en cuenta que debes implementar los getters, setters y sobrescribir el método *toString()*. En caso de que alguna clase tenga atributos de tipo *ArrayList*, debes implementar también los métodos responsables de añadir y borrar elementos a la lista.

Paso 2. Métodos extras a implementar

a) Modifica el constructor de *Invitados* para que llame al *set* de *fecha_visita* con **una fecha que le vamos a preguntar al usuario** para guardar cuándo tienen que ir al *Programa*.

Usa estas instrucciones como guía:

```
//crea un LocalDate a partir de un año, mes y día
LocalDate fecha = LocalDate.of(2025, 03, 15);
//con el formato montado, ya podemos usarlo en constructores o métodos que esperen datos LocalDate
Persona persona = new Persona(fecha);
```

b) Método ***invitadosTemporada(int temporada)*** que muestre **cuántos *Invitados*** han acudido al *Programa* dada una temporada. Muestra también sus ***nombres*** y ***profesiones***.

c) Método ***int vecesInvitado(String nombre)*** que devuelva las veces que ha ido un Invitado al *Programa*.

d) Método ***rastrearInvitado(String nombre)*** que haga uso del método creado en el apartado anterior e imprima las veces que ha ido un Invitado al *Programa*, además de mostrar también en qué ***fechas*** y ***temporadas***.

e) Método **boolean *buscarInvitado(String nombre)*** para que dado un *Invitado* sea capaz de buscar si ha acudido a un *Programa*.

f) Método ***invitadoAntes(String nombre)*** que use el método implementado en el apartado anterior para que en caso de haber devuelto *true* buscando en dos *Programas* distintos, muestre en cuál ha estado antes.

Usa estas instrucciones como guía:

```
//dadas dos fechas
LocalDate fecha1 = LocalDate.of(2023, 5, 10);
LocalDate fecha2 = LocalDate.of(2024, 1, 31);
//las comparamos para saber cuál es mayor
if (fecha1.isBefore(fecha2)) {
    System.out.println("fecha1 es ANTES que fecha2");
}
```

Ejemplo de funcionamiento:

```
Cadena [nombre='Antena 3', listaProgramas=[]]

Programa{nombre='El Hormiguero', cadena=Antena 3, director=Empleado{nombre='Director1', id='EP001', cargo='director', director=null},
temporadas=0, listaEmpleados=[Empleado{nombre='Director1', id='EP001', cargo='director', director=null}], listaInvitados=[]}

 Cadena [nombre='Antena 3', listaProgramas=[Programa{nombre='El Hormiguero', cadena=Antena 3, director=Empleado{nombre='Director1',
 id='EP001', cargo='director', director=null}, temporadas=0, listaEmpleados=[Empleado{nombre='Director1', id='EP001', cargo='director',
 director=null}], listaInvitados=[]}]]

Programa{nombre='El Hormiguero', cadena=Antena 3, director=Empleado{nombre='Director1', id='EP001', cargo='director', director=null},
temporadas=0, listaEmpleados=[Empleado{nombre='Director1', id='EP001', cargo='director', director=null}, Empleado{nombre='Pablo Motos',
id='EP002', cargo='presentador', director=Empleado{nombre='Director1', id='EP001', cargo='director', director=null}}], listaInvitados=[]}

[Empleado{nombre='Director1', id='EP001', cargo='director', director=null}, Empleado{nombre='Pablo Motos', id='EP002',
cargo='presentador', director=Empleado{nombre='Director1', id='EP001', cargo='director', director=null}}]

Introduce el año en el que acudirá el invitado Aitana:
2025
Introduce el mes:
12
Introduce el día:
12

[Invitado{nombre='Aitana', profesion='cantante', fecha_visita=2025-12-12, temporada=1}]
```

Para el siguiente *main* de ejemplo:

```
public static void main (String[] args){

  //creamos una cadena de tv
  Cadena antena3 = new Cadena("Antena 3");
  System.out.println(antena3);

  //creamos un programa
  Programa el_hormiguero = new Programa("El Hormiguero",antena3,"Director1");
  System.out.println(el_hormiguero);
  System.out.println(antena3);

  //insertamos empleados en el programa
  el_hormiguero.insertarEmpleado("Pablo Motos","presentador",null);
  System.out.println(el_hormiguero);

  //ver empleados del programa
  System.out.println(el_hormiguero.getListaEmpleados());

  //insertamos invitados en el programa
  el_hormiguero.insertarInvitado("Aitana","cantante",1);

  //ver invitados del programa
  System.out.println(el_hormiguero.getListaInvitados());

}
```

➔ REALIZACIÓN DE LA PRÁCTICA

Sigue los siguientes pasos para realizar la práctica. **¡Ve probando tu trabajo de vez en cuando para evitar que nos volvamos locos si hubiera algún error!**

1. Programa en Java la aplicación requerida
2. Plan de pruebas. Realiza las pruebas necesarias para comprobar que el programa funciona bien
3. Diagrama UML

ENTREGA

REALIZA UN INFORME EN PDF CON LA INFO GENERADA Y LOS PASOS SEGUIDOS PARA REALIZAR ESTA PRÁCTICA. EXPLICA TU CÓDIGO.

SÚBELO TODO A LA TAREA DE AULES DISPONIBLE. ADEMÁS, PEGA LA URL DE TU PROYECTO EN GITHUB.