FUNDAMENTOS DE PROGRAMACIÓN

Prácticas de Laboratorio: Diseño de tipos

En este primer laboratorio, vamos a practicar los conceptos vistos en clase de teoría, y algunos que seguramente no se habrán dado todavía. En este último caso, se pide al alumno que sea proactivo, y que trate de buscar la información en la Red.

Para la transformación de String a otros tipos, necesitará saber:

Para LocalDate:
LocalDate.parse(texto, DateTimeFormatter.ofPattern("d-M-yyyy")); previa importación (import java.time.LocalDate;). Donde 'd' es el día con uno o dos dígitos, 'M' es el mes con uno o dos dígitos, y 'yyyy' es el año con cuatro dígitos.

Para Duration:
Duration.ofMinutes(minutos).plusSeconds(segundos); previa importación (import java.time.format.DateTimeFormatter;). Donde 'minutos' y 'segundos' deben ser de tipo Long.

Para Long:
Long.parseLong(texto);

Para un tipo enumerado:
Tipo.valueOf(texto);

Descripción de tipos

Implementa y prueba los siguientes tipos:

1. Cancion

Paquetes: fp.tipos.musica, fp.tipos.musica.test

Implementación: record

Propiedades:
- *titulo*, de tipo String, consultable.
- *artista*, de tipo String, consultable. Representa al intérprete de la canción.
- *duracion*, de tipo Duration, consultable.
- *fechaLanzamiento*, de tipo LocalDate, consultable.
- *genero*, de tipo enumerado Genero, consultable. Puede tomar los valores: POP, ROCK, FOLK.
- *formatoCorto,* de tipo String, consultable. Cadena que representa una canción con el siguiente formato: el título de la canción, seguido del artista entre paréntesis y la duración, por ejemplo, "Whole Lotta Love (Led Zeppelin) 3:20

Constructores y factorías:
- C1: Constructor canónico para controlar las restricciones
- F1: Factoría. Recibe una cadena con cada propiedad separado por comas, y en el mismo orden que en el apartado anterior.

Restricciones:
- R1: el valor en segundos de la duración de una canción siempre es mayor o igual que cero.

Criterio de ordenación:

las canciones se ordenan por artista y título. En el caso de coincidir artista y título, miraremos si las dos canciones comparadas son iguales (equals) en cuyo caso, devolveremos 0. Si aun coincidiendo artista y título, las dos canciones no son iguales, devolveremos un número entero aleatorio distinto de 0 (por ejemplo, 1)

2. ListaReproduccion

Paquetes: fp.tipos.musica, fp.tipos.musica.test
Implementación: clase

Propiedades:

- *nombre*, de tipo String, consultable y modificable.
- *canciones*, de tipo List<Cancion>, consultable
- *numCanciones*, de tipo Integer, consultable. Se calcula a partir de la propiedad anterior.

Constructores:

- C1: recibe un parámetro para indicar el nombre de la lista. Al crearse, la lista de reproducción no contendrá canciones.
- C2: recibe un parámetro por cada propiedad básica

Representación como cadena:

- el nombre de la lista, seguido del número de canciones entre paréntesis. Por ejemplo: "Música tranquila (142 canciones)".

Criterio de igualdad:

dos listas de reproducción son iguales si lo son todas sus propiedades básicas.

Otras operaciones:

- void incorpora(Cancion c): añade la canción al final de la lista de reproducción. Si la canción ya estaba, se lanza IllegalArgumentException.
- void elimina(Cancion c): elimina la primera aparición de la canción en la lista de reproducción. Si la canción no pertenece a la lista, se lanza IllegalArgumentException.
- Cancion consulta(Integer i): Devuelve la canción de la posición i-ésima. Si la posición no está comprendida entre 0 y numCanciones-1, se lanza IllegalArgumentException.
- void ordena(): Ordena la lista de canciones. Puede usar Collections.sort() previa importación (import java.util.Collections;)