Programación Orientada a Objetos
Curso 2023/2024

Sesión 10

Programación Funcional

Ejercicio 1

Previo: utiliza un generador de números aleatorios (`java.util.Random`) para crear una **lista** con 20 números aleatorios cuyos valores estén comprendidos entre 0 y 99.

Utiliza el procesamiento *stream* sobre la colección de números aleatorios y las expresiones lambda para implementar la siguiente funcionalidad:

- Muestra por la consola la representación en hexadecimal de los números de la lista.

  Nota: El método `Integer.toHexString` convierte un entero en su representación hexadecimal.
- Igual que el anterior, pero mostrando las cadenas hexadecimales ordenadas alfabéticamente.
- Muestra por la consola la cantidad de números que son pares.
- Construye un mapa que asocie cada número de la lista con su representación hexadecimal.
- Consulta si hay algún número en el rango de valores 55 – 60. Muestra el resultado por la consola.

Ejercicio 2

Previo a los ejercicios

Implementa la clase que representa un registro de acceso como sigue:

```
public class Registro {
	private final String usuario;
	private final LocalDate entrada; //fecha de entrada
}
```

Genera automáticamente los métodos `get` para todas las propiedades y el constructor que inicialice todas las propiedades con los valores que se establezcan como parámetro.

La clase **`LocalDate`** es una clase `Comparable` que se encuentra en el paquete `java.time` y dispone de los métodos de clase `now()`, que devuelve la fecha actual, y `of(int año, int mes, int dia)` que construye un objeto de a partir de la información pasada como parámetro. Por ejemplo, para crear la fecha 11/12/2023 sería `LocalDate.of(2023,12,11)`.

Además, dispone de los métodos `isAfter(LocalDate otra)` e `isBefore(LocalDate otra)` para comparar si una fecha es posterior o anterior (respectivamente) a la fecha que se pasa como parámetro.

Crea los objetos de tipo `Registro` y almacénalos en una **lista**:

- Pedro accede el 11/11/2023;
- Juan accede el 4/11/2023;
- Martina accede el 12/11/2023;
- Andrea accede el 4/11/2023;
- Pedro accede el 12/11/2023.

1. Utiliza el procesamiento *stream* sobre la colección de registros y las expresiones lambda para implementar la siguiente funcionalidad:

- Obtener un conjunto con todos los usuarios que han realizado algún acceso desde 10/11/2023, no incluido.
- Mostrar el nombre de los usuarios que han accedido el día 12/11/2023 de forma alfabética.
- Contar el número de accesos desde el 4/11/2023 y hasta 12/11/2023, ambos incluidos. Mostrar el resultado en la consola.
- Consultar si algún usuario ha realizado un registro el día 4/11/2023. Mostrar el resultado en la consola.

2. Implementa un método de clase **(dolf)** que aplique una acción (`Consumer`) sobre todos los registros de una lista que cumplan una determinada condición (`Predicate`) haciendo uso de un iterador explícito. Utiliza el método anterior para mostrar en la consola todos los registros del 12 de noviembre de 2023.

3. Generaliza el método anterior para que se pueda aplicar sobre cualquier colección de objetos.