Ejercicio Sesión 04

La práctica corresponde al

Tema 03: Estructuras de Datos Lineales

Objetivos

Con esta práctica se pretende que el alumno se familiarice con el uso de las estructuras lineales que se proporcionan de forma predeterminada en el *Java Collections Framework*. Son las agrupadas bajo los interfaces `List` y `Queue`.

Adicionalmente se trabajará con la Genericidad de Java, que permite definir los tipos de datos a almacenar en una colección. Es una característica que mantienen todos los componentes del *Collections Framework*.

Problema planteado

Se nos facilita un log de acciones que se han ido recogiendo a lo largo de la sesión de un usuario sobre nuestro sistema. Por simplicidad, cada acción se representará en nuestro caso solo por la fecha y hora del mismo. Se quiere ahora agrupar los eventos concretos que se pueden localizar en esa sesión. Un evento concreto se determina porque el intervalo de tiempo entre una acción y la siguiente será corto (y se podrá determinar por el usuario).

Se pide al alumno construir una clase de nombre SeparadorFechas.java. En su interior tendrá un solo método estático con la signatura:

```
public static Deque<List<LocalDateTime>> separaFechas
(Deque<LocalDateTime> listaFechas, Duration maxSeparacion)
```

A este método se le pasa como parámetros:

- `Deque<LocalDateTime> listaFechas`: una cola de instantes de tiempo (fecha y hora) consecutivas en el tiempo. Por su condición de cola, estas no se pueden desordenar o extraer en orden alterno.
- `Duration maxSeparacion`: la duración máxima entre fechas para considerarlo como parte del mismo evento. Todas aquellas fechas que tengan un espacio entre ellas igual o menor a esta duración se consideran agrupadas en el mismo evento. Al ser consecutivas en el tiempo, solo se necesita comparar una fecha con su inmediatamente posterior en la lista. En Java 8 el periodo de tiempo entre dos fechas se puede calcular como:

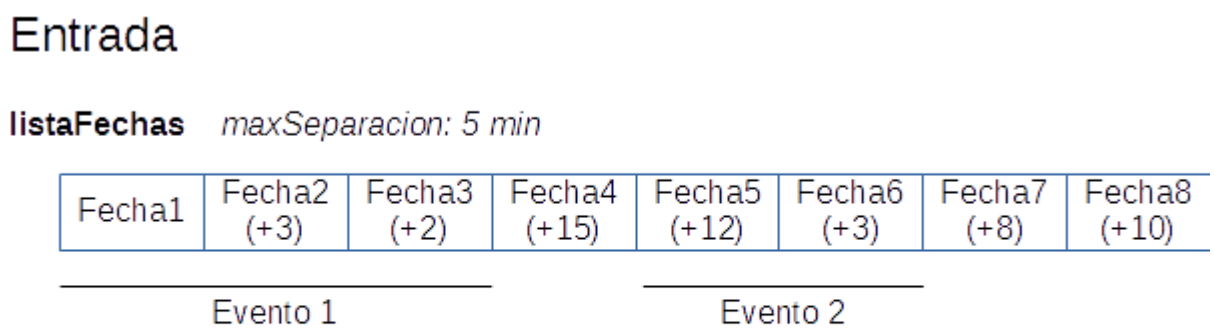
  ```
  Duration d = Duration.between(DateTime1, DateTime2);
  ```

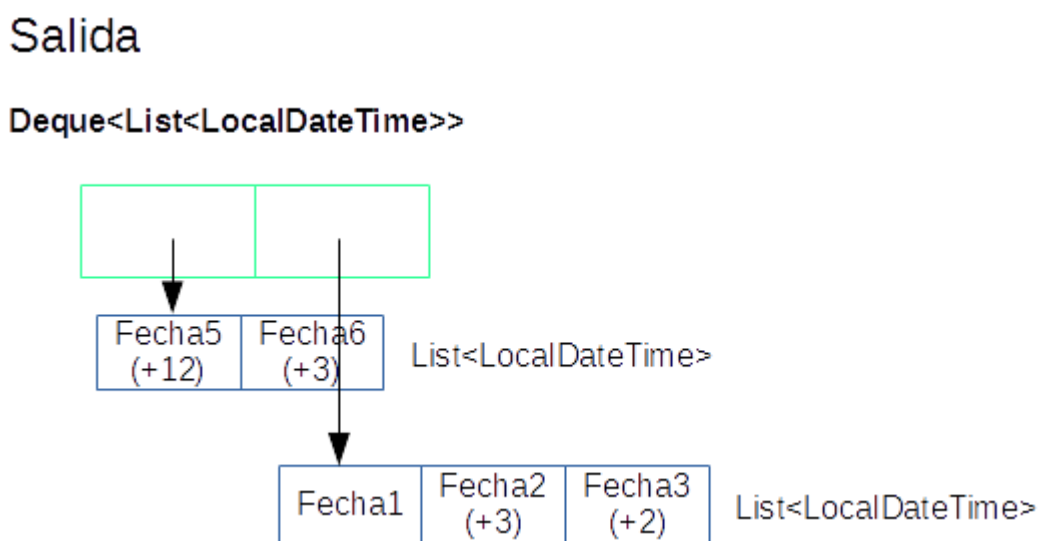
  La clase `Duration` implementa el interfaz `Comparable`, por lo que se puede saber si una duración es mayor o menor que otra con el método `compareTo()`.

Se organizarán los eventos de forma que todas las fechas y horas que correspondan al mismo evento se incluirán en una misma lista.

Se pide devolver el conjunto de los eventos encontrados (es decir listas de `LocalDateTime`) en una colección que contenga listas de solo 2 fechas: la inicial y la final del evento. Dentro de la colección de tipo Deque (DobleCola) que se devuelve, se pide organizar del evento **más reciente** (en la cabeza de la cola), compuesto por el par de fechas más reciente hasta el **más antiguo** (en

la cola de la cola). Para ello lo más sencillo es emplear una Cola de tipo `Deque` que permita ir almacenando cada lista correspondiente a un evento según se vaya localizando en la lista inicial, insertando siempre por el principio.





Criterios de Evaluación

Dada la naturaleza y objetivo de la práctica, se tendrá en cuanta para su valoración (de más importante a menos):

1. La corrección del funcionamiento. Se valorará como aspecto más importante que el funcionamiento de la estructura de datos implementada cumpla con todas las condiciones solicitadas, tanto en la documentación oficial como en el enunciado.
2. La complejidad algorítmica de la solución aportada. Se deberá tener cuidado de implementar la respuesta por medio de soluciones que impliquen la menor complejidad siempre que esto sea posible.
3. La corrección del código. Ausencia de cosas como variables no empleadas, código inalcanzable, etc.
4. La documentación y comentarios que incluya el alumno en el código entregado.

Condiciones de Entrega

La práctica se realizará en grupos de 1 o 2 personas.

La entrega ser hará SOLAMENTE por medio de la plataforma UBUVirtual. Cada miembro del grupo deberá subir su propia copia de la solución. No se admitirán soluciones fuera de plazo.

La entrega incluirá la información de los miembros de la pareja de prácticas, tanto en los ficheros de código fuente como en los documentos asociados.

Cada entrega consistirá en un fichero comprimido (formato .zip), con la estructura de nombre: `Apellidos1Nombre1_Apellidos2Nombre2.zip`

Deberá que incluir dentro:

- Código fuente de la solución (dentro de la estructura de paquetes necesaria)
- Comentarios dentro del código: El fichero fuente deberá estar debidamente comentado con todas las aclaraciones necesarias para la mejor comprensión de la solución.

No hace falta entregar ficheros binarios.