Sesión 2. Iteradores

Crea un proyecto Java de nombre, `sesión-02` y, si es necesario, cambia la codificación de caracteres a UTF-8.

Material proporcionado para la práctica (descarga del CV y descomprime el archivo *sesión-02.zip*):

1. La interfaz *PeekIterator<E>* que extiende la interfaz Java *Iterator<E>*.
2. La clase *Sucesion()* disponible también en las diapositivas de teoría y que permite generar los términos de la sucesión: `0`, `1`, `3`, `6`, `10`, `15`, `21`, … En esta versión se ha cambiado el iterador para trabajar con objetos de tipo *Integer* y, además, la operación *hasNext()* siempre retorna *true* (generación potencialmente indefinida).
3. Un programa de ejemplo que solicita un entero no negativo, *limit*, y muestra los elementos de la sucesión que son menores o igual a *limit* (valor umbral), así como la suma de éstos. En este caso, para obtener los términos de la sucesión se utiliza la función *nthTerm(n)* que retorna su término enésimo.

La interfaz *PeekIterator<E>*

Como se ha visto en clase de teoría y especifica la API de Java, la operación *next()* de la interfaz *Iterator<E>* retorna un elemento de la iteración y avanza el cursor al siguiente elemento. Esto, en ocasiones, puede resultar inconveniente. Así ocurre, por ejemplo, cuando se necesita conocer cuál es el siguiente elemento de la iteración para hacer efectiva la iteración correspondiente más adelante (con la operción *next()* se obtiene el elemento y se realiza la iteración).

Con la interfaz *PeekIterator<E>* se pretende solucionar el posible inconveniente antes citado. Para ello, esta interfaz especifica la operación adicional *peek()* (echar un vistazo) que permite conocer el elemento a iterar sin realizar la iteración (el cursor, desde el punto de vista de su uso, no avanza al siguiente elemento).

Se pide:

1. Proporcionar la clase *PeekingIterator<E>* que implementa la interfaz *PeekIterator<E>* en base a las operaciones de la interfaz *Iterator<E>* que extiende. El constructor de la clase recibirá un objeto iterable.

   Nota: antes de implementar la operación *remove()* consulta la especificación de esta operación en la interfaz *Iterator<E>*.

2. Añade al programa el código necesario para volver a obtener y mostrar los términos de la sucesión que son menores o iguales que *limit*, así como la suma de éstos, pero ahora utilizando un iterador de la clase previamente realizada (*PeekingIterator<E>*). Los resultados deberán mostrarse exactamente igual que en el caso inicial proporcionado y para obtener éstos únicamente podrá utilizarse la variable *total* para la suma de los términos y las operaciones de la clase.

   Los resultados obtenidos deberían ser los mismos, pero ¿cuál es la diferencia entre obtener los términos con la función *nthTerm(n)* y obtenerlos con el iterador?