Programación Orientada a Objetos
Curso 2023/2024

Sesión 4

Previo

Para la implementación de los ejercicios es necesario hacer uso de listas. En Java existen varias clases que implementan listas. En esta práctica utilizaremos la clase `java.util.LinkedList`. Ten en cuenta que las colecciones de Java son clases genéricas. La genericidad se estudia más adelante en la asignatura. Para la implementación de la práctica es suficiente saber cómo se declaran y construyen las listas:

```
LinkedList<Subasta> subastas = new LinkedList<Subasta>();
```

Observa que se incluye el tipo de elementos que va a contener la lista entre **<** y **>**. En el ejemplo anterior se ha utilizado el constructor sin argumentos pero también está disponible el constructor de copia.

La documentación de la clase `java.util.LinkedList` está disponible en http://docs.oracle.com/javase/8/docs/api/java/util/LinkedList.html o línea en Eclipse.

A continuación, se muestra un fragmento de código con las operaciones básicas para el manejo de listas:

```
LinkedList<String> lista = new LinkedList<String>();

// Añadir elementos por el final
lista.add("uno");       // Opción 1
lista.addLast("dos");   // Opción 2

// Añadir un elemento al principio
lista.addFirst("cero");

// Consultar el tamaño
System.out.println("Tamaño: " + lista.size());

// Consultar el primer elemento
String primero = lista.get(0);      // Opción 1
primero = lista.getFirst();         // Opción 2
System.out.println(primero);

// Consultar el último elemento
String ultimo = lista.get(lista.size() - 1);    // Opción 1
ultimo = lista.getLast();                       // Opción 2

System.out.println(ultimo);

// Borrado: remove, removeFirst y removeLast

// Recorrido for each
for (String elemento : lista) {
      System.out.println("Elemento: " + elemento);
}
// Copia
LinkedList<String> copia = new LinkedList<String>(lista);
```

Ejercicios

Crea el paquete `subastas` y sitúa en ese paquete todo el código de esta práctica. El objetivo de los ejercicios es implementar un sistema de **subastas** donde los usuarios puedan pujar por productos que ofrecen otros usuarios. A continuación, se describen los requisitos de la aplicación.

1. Define la clase **`Puja`** de la aplicación de subastas. Una puja representa la cantidad de dinero que ofrece un usuario.

   Las propiedades que caracterizan a una Puja son (todas consultables):

   - pujador: cadena de texto con el nombre del usuario que hace la puja.
   - cantidad: número real que representa la cantidad de dinero que se ofrece en la puja.

   Las propiedades no pueden ser modificadas. Como consecuencia, la clase Puja permite construir objetos *inmutables* que se pueden compartir sin que el aliasing comprometa la información.

   Las pujas se construyen estableciendo las dos propiedades que caracterizan a la clase.

   Por último, la clase no proporciona ninguna funcionalidad.

2. Define la clase **`Subasta`** que representa un producto por el que se pueden realizar pujas.

   Las propiedades que caracterizan a una subasta son (todas consultables):

   - nombre del producto: cadena que describe el producto subastado.
   - propietario: nombre del usuario que es propietario de la subasta.
   - abierta: valor booleano que indica si la subasta está abierta o cerrada. El valor `true` significa que está abierta. En cambio, el valor `false` indica que está cerrada.
   - pujas: lista de pujas realizadas por el producto que ofrece la subasta.

     Nota: los objetos de tipo Puja se pueden compartir sin riesgo. Sin embargo, la lista que contiene todas las pujas no debe ser compartida. Así pues, el método de consulta de esta propiedad debe realizar una *copia defensiva* de la lista para evitar el riesgo de aliasing.

   - puja mayor: referencia al objeto Puja con la cantidad mayor ofrecida en la subasta (*propiedad calculada*). En caso de que no existan pujas, el valor de esta propiedad es `null`.

   No se ofrecen operaciones directas de modificación (métodos *set*) sobre las propiedades. Las propiedades cambiarán aplicando la funcionalidad que ofrece la clase.

   Una subasta se construye estableciendo el nombre del producto subastado y el nombre del usuario propietario. Una subasta recién construida está abierta y no tendrá pujas.

La funcionalidad de esta clase es:

- *pujar*: este método permite realizar una puja sobre la subasta.

  Los parámetros de este método son: el nombre del usuario que realiza la puja y la cantidad por la que puja.

  La puja es aceptada si: a) la subasta está abierta, b) el usuario no es propietario de la subasta y c) la cantidad es mayor que la cantidad de la *puja mayor*, si la hubiera.

  Si la puja es aceptada, entonces se construye una puja y se almacena en la lista de pujas.

  Por último, esta operación finaliza indicando si la puja ha sido aceptada (retorna un valor booleano).

- *pujar*: versión sobrecargada del método anterior que permite pujar sin indicar la cantidad. Es decir, sólo se tiene como parámetro el nombre del usuario que la realiza.

  La cantidad será un euro más que la cantidad de la *puja mayor*. Si no hubiera *puja mayor*, la cantidad sería de un euro.

- *ejecutar*: este método cierra la subasta. Esta operación no tiene parámetros.

  Una subasta se puede ejecutar si se cumplen las siguientes condiciones: a) existe alguna puja y b) la subasta está abierta.

  La ejecución de una subasta consiste en establecer a falso el valor de la propiedad abierta.

  El método finaliza informando si la subasta ha podido ejecutarse o no (retorna un valor booleano).

3. Escribe el siguiente programa:
   - Crea una subasta del producto "Teléfono Móvil" cuyo propietario sea el usuario “Juan”.
   - El usuario “Pedro” puja por esa subasta 100 euros.
   - Muestra en la consola la puja mayor de la subasta (nombre del usuario y cantidad).
   - El usuario “Enrique” puja por esa subasta 50 euros.
   - Muestra en la consola la puja mayor. Comprueba que esta segunda puja no ha sido acepta, ya que es menor que la primera.
   - Ejecuta la subasta.
   - El usuario “Enrique” puja de nuevo por esa subasta con 200 euros. Comprueba que no es aceptada, ya que la subasta ha sido cerrada.