Programación Modular y Orientación a Objetos - Laboratorio 7

Requisitos previos

El alumnado es capaz de resolver problemas que requieren implementar varias clases relacionadas, entre las que hay tanto MAEs como TADs, y realizar las pruebas unitarias correspondientes con JUnit. Además está familiarizado con los diagramas de clase y de secuencia UML, y comprende a nivel teórico el concepto de herencia simple entre clases, así como las ventajas que aporta a nivel de reusabilidad.

Objetivos

El objetivo de este laboratorio consiste en trabajar los aspectos de herencia vistos en clase, así como afianzar los conocimientos adquiridos en laboratorios anteriores, especialmente los relativos a la programación orientada a objetos como alternativa a la programación imperativa, y al diseño y a la creación de diagramas de clase y de secuencia.

Al finalizar este laboratorio, el alumnado deberá ser capaz de:

- Diseñar la solución de problemas que requieran la definición de varias clases (MAEs y TADs) interrelacionadas y que incluyan herencia simple.
- Dibujar diagramas de clases y de secuencia sencillos.
- Implementar varias clases interrelacionadas (MAEs y TADs), así como sus correspondientes casos de prueba con JUnit.

Motivación

Este laboratorio vuelve a presentar un ejercicio de cierta complejidad, que en este caso trabaja además la herencia simple. Aunque inicialmente el enunciado no facilita ningún diseño de la solución a implementar, quedando en manos del alumnado su descripción (en términos de diagramas de clase y de secuencia), después se proporciona un diagrama de clases con objeto de unificar las soluciones entregadas por los alumnos.

Tarea única: parejas de baile estables

Comienza un nuevo curso en la escuela de bailes de salón. Los alumnos están organizados en parejas, pero de un año a otro se desea rehacerlas porque siempre hay personas que prefieren cambiar de pareja.

En los bailes de salón se respetan ciertas reglas de protocolo. Así, las damas de más edad tienen el privilegio de comenzar a elegir pareja de baile. Por este motivo, las parejas de alumnos siempre están ordenadas de mayor a menor edad de la dama. Por otro lado, se ha pedido a todos los alumnos (sean hombres o mujeres) que faciliten una lista con las personas del sexo opuesto con las que les gustaría formar pareja. En la lista de preferencias de una mujer no tienen por qué estar todos los hombres del curso, y viceversa; pero cada alumno debe incluir al menos a una persona del sexo opuesto, que podría ser su pareja actual si es que está a gusto con ella.

Usando las listas de preferencias, se quiere encontrar un emparejamiento estable, es decir, un reajuste de las parejas en el que no se den situaciones como la siguiente:

```
Miren está emparejada con Aitor.
Beñat está emparejado con Saioa.
Miren prefiere estar con Beñat y Beñat prefiere estar con Miren.
Saioa prefiere estar con Aitor y Aitor prefiere estar con Saioa.
```

Para ello, se seguirá el siguiente algoritmo, que tratará de reajustar las parejas para obtener un emparejamiento estable, y que si no lo consigue dejará las parejas como estaban al comienzo:

Paso 0: En primer lugar, se verificará que todos los alumnos (hombres o mujeres) del curso tienen una pareja asignada. Si existiera algún alumno sin pareja, el proceso de reasignación de parejas no se realizaría, y en su lugar se mostraría un mensaje por pantalla avisando de tal circunstancia. Si no, se procedería con el paso 1.

Paso 1: Crear una lista de hombres disponibles, que inicialmente incluirá a todos los alumnos masculinos de la escuela. Dado que previamente se ha comprobado que no hay alumnos (y por tanto hombres) sin pareja, la lista de todos los alumnos masculinos puede generarse directamente a partir de la lista de parejas.

Paso 2: Definir una variable auxiliar que permita almacenar una lista de parejas, que inicialmente estará vacía.

Paso 3: Recorrer la lista de parejas del curso (que estará ordenada de mayor a menor edad de la mujer, dado que la inserción de parejas se habrá realizado en su momento mediante el método *anadirOrdenadoPareja()*), hasta que no se pueda continuar:

Paso 3.1: Recorrer las preferencias de la dama de la pareja actual hasta que no se pueda seguir, o se le haya asignado una pareja.

Al mirar una preferencia de la dama podrían suceder dos cosas:

Situación 3.1.a: que no se le pueda asignar la preferencia de la dama como pareja, bien porque el hombre no está en la lista de disponibles (porque una mujer de más edad lo ha seleccionado antes) o bien porque el hombre no ha incluido a la dama entre sus propias preferencias. En cualquiera de los dos casos, se pasaría a

mirar el siguiente hombre en la lista de preferencias de la mujer, siempre que queden preferencias por comprobar. Si ya no quedasen más hombres en la lista de preferencias de la mujer de la pareja actual, se mostraría un mensaje por la consola del sistema indicando que no se han podido reajustar las parejas, y todo el proceso terminaría.

Situación 3.1.b: que se le pueda asignar la pareja que ella desea, esto es, que su preferencia actual esté disponible y además tenga a la dama entre sus propias preferencias. En este caso habría que añadir a la lista de parejas auxiliar la pareja formada por la mujer de la pareja actual y el hombre disponible que ella desea; retirar a dicho hombre de la lista de hombres disponibles; y pasar a la siguiente pareja de la lista de parejas del curso.

Paso 4: Si se ha llegado hasta este punto (es decir, si se han podido reajustar todas las parejas), asignar a la lista de parejas del curso la lista de parejas auxiliar.

En el aula de teoría se pide:

- Dibujar el diagrama de clases y el diagrama de secuencia necesarios para realizar una correcta implementación de la solución.

Como parte del laboratorio 7 se pide:

- Implementar las clases necesarias para resolver la tares, ajustándose al siguiente diagrama de clases, y a la especificación proporcionada en los ficheros de ayuda .java que acompañan a este enunciado. NOTA: Se pide implementar una solución que se ajuste a este diagrama de clases, no al diagrama que cada alumno o grupo de alumnos haya realizado en el aula de teoría.
- Implementar los casos de prueba JUnit necesarios para verificar la corrección de estas clases.

Todas las clases deben estar dentro del paquete **org.pmoo.packlaboratorio7**.

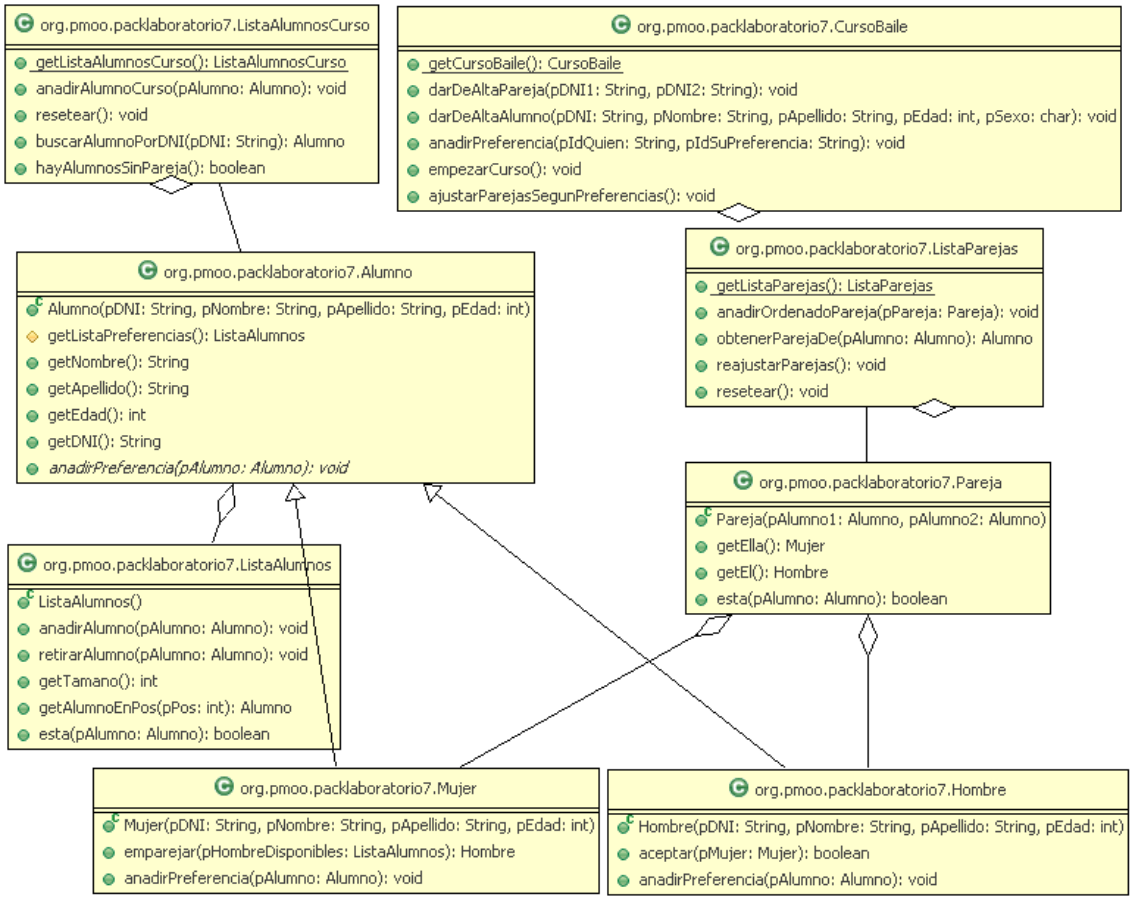

Figura 1 - Diagrama de clases para el ejercicio del Curso de Baile.