OBJETIVOS

- Diseño de tipos: criterios de igualdad y ordenación.
- Casos de prueba.

IMPLEMENTACIÓN DE LOS CRITERIOS DE IGUALDAD Y ORDENACIÓN

Complete las clases con los métodos `equals()`, `hashCode()` y `compareTo()`, teniendo en cuenta lo que sigue a continuación.

Se entenderá que dos objetos distintos del sistema representan a la misma **asignatura** si ambos objetos almacenan el mismo código. El sistema almacena las asignaturas ordenadas alfabéticamente por su código (se considera éste el orden natural para las asignaturas).

Con respecto a las **becas**, se entenderá que dos objetos distintos del sistema representan a la misma beca si ambos objetos almacenan el mismo código y el mismo tipo de beca. El sistema necesita mantener las becas ordenadas, generalmente en orden alfabético según el código; al ordenar becas que tienen el mismo código, se colocan en primera posición las becas de tipo ordinario, luego las de movilidad y por último las de empresa (se considera éste el orden natural para las becas).

Dos **personas** son iguales si tienen el mismo DNI, nombre y apellidos. Las personas se ordenan por apellidos, a igualdad de apellidos por nombre, y a igualdad de nombre por DNI.

Para que dos **espacios** sean iguales deben coincidir su nombre y su planta. Los espacios se ordenan por planta, y a igualdad de ésta por nombre.

Dos **notas** son iguales si corresponden al mismo curso académico, la misma asignatura y la misma convocatoria. Las notas se ordenan por curso académico, a igualdad de curso por asignatura, y a igualdad de asignatura por convocatoria.

En cuanto a las **tutorías**, se consideran iguales si se realizan el mismo día y tienen la misma hora de comienzo. Para ordenarlas se considera el día, y a igualdad de éste la hora de comienzo.

CASOS DE PRUEBA

Implemente un test para cada tipo donde compruebe el correcto funcionamiento de los nuevos métodos creados. La metodología que se utilizará para probar el criterio de igualdad y el de ordenación será la que se explica a continuación.

Para probar la igualdad de un tipo, cree dos objetos del tipo con el mismo valor en las propiedades que participan en el criterio de igualdad, y distintos valores en el resto de propiedades. Compruebe que ambos objetos son iguales según el método `equals()`, y que el método `hashCode()` de ambos devuelve el mismo número. Cree posteriormente objetos distintos a los anteriores, variando los valores

de las propiedades que participan en el criterio de igualdad (tantos objetos distintos como propiedades intervengan en el criterio). Compruebe que estos objetos son distintos según el método `equals()`[1].

Para probar la relación de orden de un tipo, cree cuatro objetos del tipo: uno menor, dos iguales entre sí y mayores que el anterior, y uno mayor que todos los anteriores. Compruebe comparándolos entre ellos que el método `compareTo()` funciona correctamente.

[1] Es deseable que el método `hashCode()` de ambos objetos devuelva valores distintos, aunque no es estrictamente necesario.