Programación orientada a objetos
Relación de ejercicios

1. Proponga tres ejemplos de objetos del mundo real:

    - Para cada uno de ellos, determine la clase a la que pertenecen.
    - Asóciele a cada clase un identificador descriptivo adecuado.
    - Enumere varios atributos y operaciones para cada una de las clases.
    - Represente gráficamente las clases utilizando la notación UML.
    - A partir de los diagramas UML, escriba el código necesario para definir las clases utilizando el lenguaje de programación Java.

2. Rellene los huecos en las siguientes afirmaciones:

    a. Los objetos encapsulan _____ y _____.

    b. Los objetos se comunican entre sí pasándose _____.

    c. Para comunicarse con un objeto concreto, no es necesario conocer su _____, basta con saber cuál es su _____.

    d. Pueden existir varios tipos de relaciones entre clases: _____, _____ y _____.

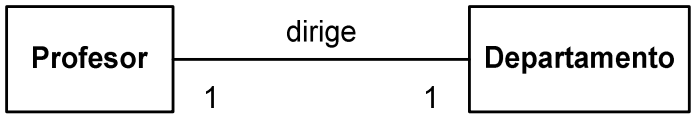
    e. Los lenguajes de programación orientada a objetos utilizan relaciones de _____ para derivar nuevas clases a partir de clases base.

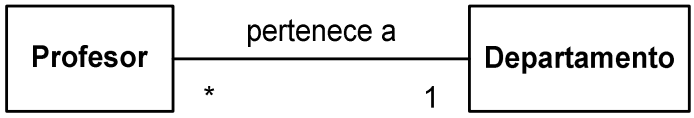
    f. _____ define una notación gráfica estándar para representar diseños orientados a objetos.

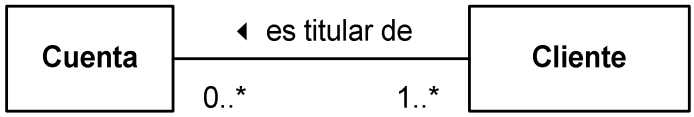
    g. Las clases se definen en Java en ficheros de texto con la extensión _____

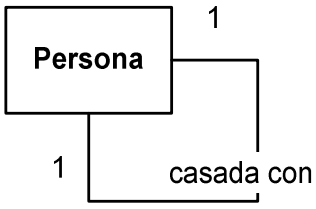
    h. El compilador de Java genera ficheros con extensión _____ al compilar un fichero de código fuente escrito en Java.

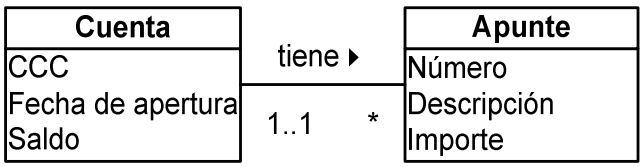
3. Definir adecuadamente las clases en Java que se derivan de los siguientes diagramas de clases UML:

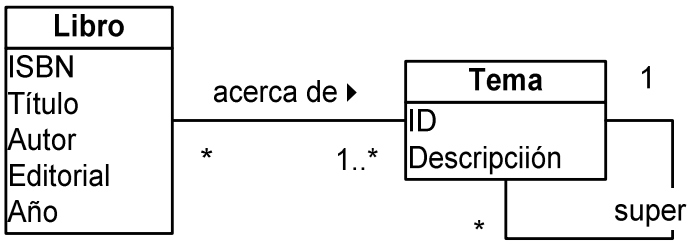


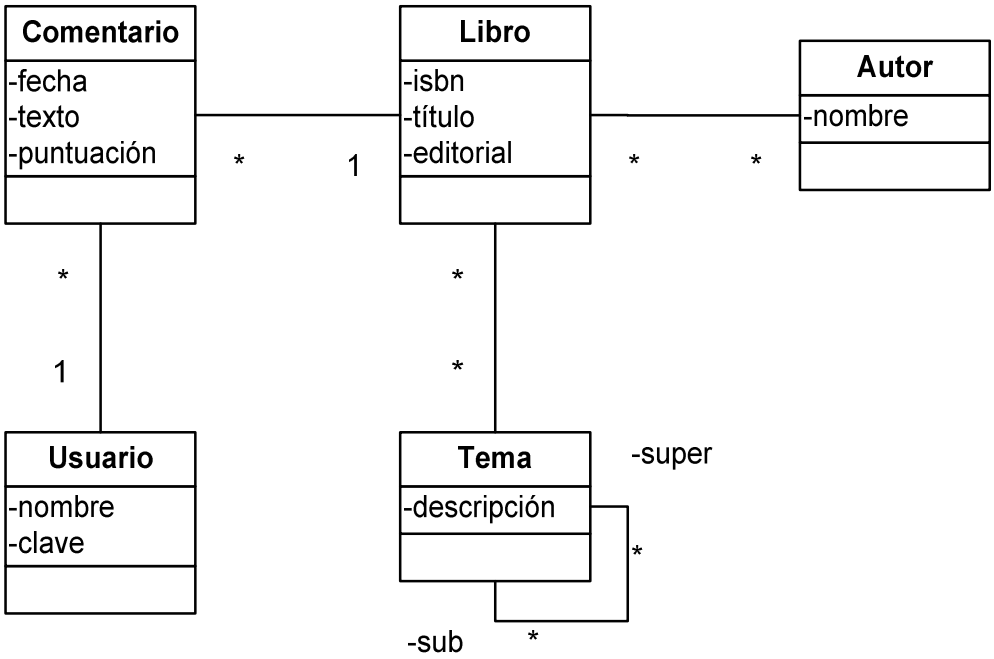


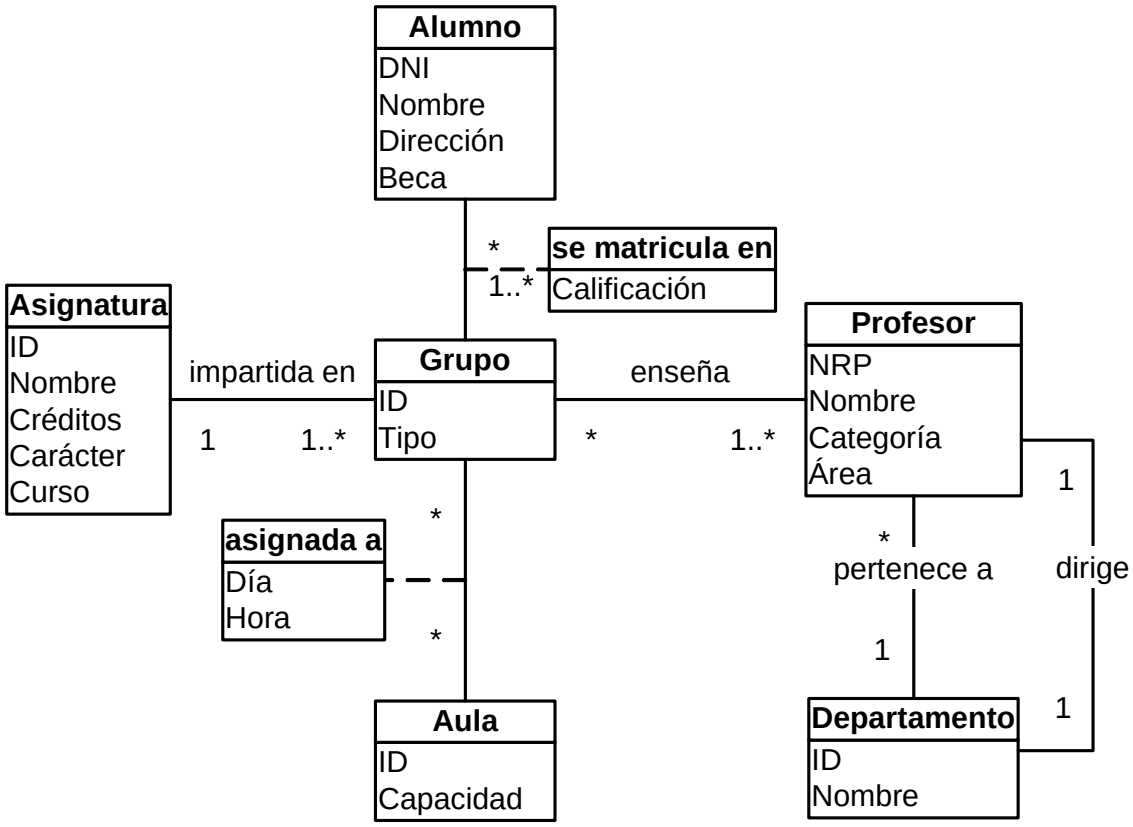






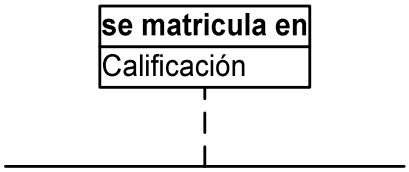



Comentario
-fecha
-texto
-puntuación
Libro
-isbn
-título
-editorial
Autor
-nombre
Usuario
-nombre
-clave
Tema
-descripción
-super
-sub
*
1


Alumno
DNI
Nombre
Dirección
Beca
se matricula en
Calificación
Asignatura
ID
Nombre
Créditos
Carácter
Curso
impartida en
Grupo
ID
Tipo
enseña
Profesor
NRP
Nombre
Categoría
Área
asignada a
Día
Hora
pertenece a
dirige
Aula
ID
Capacidad
Departamento
ID
Nombre
1
1..*
*

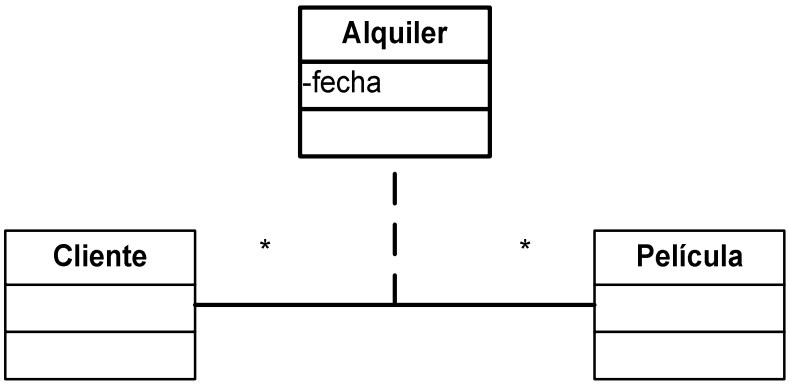
Nota: CLASES ASOCIACIÓN

Las clases asociación (como “se matricula en”) se emplean para indicar que la asociación existente entre dos clases tiene atributos propios:



En realidad, las clases asociación de un diagrama de clases UML son clases convencionales cuyo único papel consiste en relacionar objetos de otras clases (no tienen comportamiento propio)

Ejemplo



La fecha del alquiler no es un atributo del cliente ni de la película,
es algo específico del hecho de alquilar la película.

```
class Cliente
…
class Pelicula
…

class Alquiler
{
  private Cliente  cliente;
  private Pelicula peli;
  private DateTime fecha;

  public Alquiler
         (Cliente cliente, Pelicula peli, DateTime fecha)
  {
    this.cliente = cliente;
    this.peli = peli;
    this.fecha = fecha;
  }
  …
}
```