Composición y Agregación

Guía de Ejercicios

Ejercicio 1

Crear la clase **NumeroTelefonico**, la cual tiene como atributos numéricos *característica*, *numeroDeAbonado* y *codigoDePais*, además un atributo de tipo *TIpoDeLinea* que indicará el tipo de línea (CELULAR, FIJO, FAX). Generar los constructores por defecto y parametrizado y los setters y getters que creas necesarios. Tiene también un método llamado *getValor()* que devolverá un String en formato "(+PPP) CCCC-AAAA", donde +PPP es un signo más (+) seguido por el código del país.
Crear también la clase **EMail** cuyo constructor deberá recibir un único String con el formato "cuenta@dominio.del.mail". Para llenar los atributos *cuenta* y *dominio*, ambos de tipo String, se deberá usar el método *split(separador)* de la clase String, usando la arroba ("@") como separador. El método *getValor()* devolverá el String con la "reconstrucción" de la dirección, utilizando ambos atributos más la arroba.
Por último, crear la clase **Persona** (su constructor recibe *apellido* y *nombre*) podrá contener de 0 a N teléfonos y de 0 a N emails. Su método *mostrarTodo()* debe mostrar por pantalla el contenido completo de la persona. Por ejemplo:

```
Messi, Lionel
Telefonos:
Celular: 549-114111-2222
Celular: 068-032444-5678
Fijo: 054-4411-5472
email: lio@messi.com
```

Ejercicio 2

Reutilizando las clases del ejercicio anterior, extender la clase **Persona** para que pueda tener de 0 a N **Mascotas** (contiene el nombre y el tipo de animal, ambos String). Debe tener los métodos necesarios para agregar nuevos datos o eliminar cada uno de los existentes (menos los datos de la persona). Determinar si corresponde utilizar composición y agregación.
Una salida posible podría ser la siguiente:

```
Messi, Lionel
Telefonos:
Celular: 549-114111-2222
Celular: 068-032444-5678
Fijo: 054-4411-5472
Emails:
```

```
lio@messi.com
liomessi_newells@hotmail.com
Mascotas:
Perro, Pluto
Gato, Felix
Conejo, Bugs
```

Ejercicio 3

Reutilizando las clases del ejercicio anterior, agregar la clase **Hito** (con fecha, descripción, personas involucradas) para poder agregar momentos importantes en la vida de la persona (puede tener 0 a N). Un mismo hito puede ser utilizado para más de una persona.

Ejercicio 4

Crear la clase **Vivienda** que tendrá una **Dirección** (compuesta por: calle, altura, piso y departamento) y que además pueda contener **personas** (nombre, apellido y edad) y **muebles** (nombre, material y color). La salida deberá ser la siguiente:

```
Vivienda 1:
Direccion: Montañeses 1234 4°”C”
Personas:
Nombre: Arturo Roman, Edad: 53
Nombre: Mónica Gaztambide, Edad: 35
Muebles:
Mesa de Madera color Marrón.
Mesada de Mármol color Rojo
Perchero de Metal color Negro
Sillón de Cuero color Azul
```

Ejercicio 5

Reutilizando la clase **Vivienda** del ejercicio anterior, crear la clase **Edificio** que contenga una colección de viviendas. A través del método *realizarMudanza()*, que recibirá el piso y departamento de origen, y el piso y departamento de destino, se deberá trasladar personas y muebles de una vivienda a la otra. Determinar en qué casos corresponde utilizar composición y en cuales corresponde utilizar agregación. Finalmente, el método *mostrarTodo()* mostrará por pantalla el detalle completo de todas sus viviendas. La salida deberá ser:

```
Vivienda 1:
Direccion: Montañeses 1234 4°”C”
Personas:
Nombre: Arturo Roman, Edad: 53
```

```
Nombre: Mónica Gaztambide, Edad: 35
Muebles:
Mesa de Madera color Marrón.
Mesada de Mármol color Rojo
Perchero de Metal color negro
Sillón de cuero color azul
Vivienda 2:
Direccion: Montañeses 1234 2°”B”
Personas:
Muebles:
SE HA REALIZADO LA MUDANZA DE VIVIENDA 1 A VIVIENDA 2
Vivienda 1:
Direccion: Montañeses 1234 4°”C”
Personas:
Muebles:
Vivienda 2:
Direccion: Montañeses 1234 2°”B”
Personas:
Nombre: Arturo Roman, Edad: 53
Nombre: Mónica Gaztambide, Edad: 35
Muebles:
Mesa de Madera color Marrón.
Mesada de Mármol color Rojo
Perchero de Metal color negro
Sillón de cuero color azul
```

Ejercicio 6 (opcional)

Una tienda de libros recibe pedidos de sus clientes a través de un formulario en su página WEB. Cada pedido cuenta con los siguientes datos:

- Nro. de compra
- Fecha de compra
- Libro
- Cliente
- Datos del libro
    - Autor
    - Título del libro
    - Editorial
- Datos del cliente
    - Nombre y apellido
    - DNI
    - Email
    - Dirección de envío

Una vez por día la tienda procesa los pedidos pendientes y para ello recorre los pedidos realizados verificando si el stock de ejemplares del libro solicitado alcanza para cumplir con el pedido. Si hay stock se elimina el pedido del listado de pendientes y se agrega el pedido a un listado de pedidos completos; si no alcanza el stock el pedido debe quedar en la colección de pedidos pendientes.

Realizá el diagrama UML con las clases que consideres necesarias.
Desarrollá el método *procesarPedidos()* que cumple con el el comportamiento descrito para procesar los pedidos.
Escribí un programa para testear este modelo de clases usando un lote de datos que permita probar todas las opciones posibles.