Objetos que contienen objetos

Los atributos de un objeto pueden ser de diferentes tipos y, en este tema, profundizaremos en objetos que tienen una relación de composición con otros objetos y en objetos que están contenidos en arreglos.
Como ocurre en la vida diaria, existen objetos que están compuestos por otros objetos y cada uno tiene sus características y atributos, pero, en conjunto, los dos objetos tienen un propósito más grande; por ejemplo, podemos tener el objeto alumno que contiene objetos de tipo materias.

Relación de composición con otras clases

Entre las clases, existen muchos tipos de relaciones, una de ellas es la composición. Este tipo de relación es muy común porque consiste en que una clase tiene atributos que son de otra. Se dice que la relación es del tipo "tiene – un". Algunos ejemplos de este tipo de relación serían los siguientes:

- La clase Empleado tiene un atributo de tipo Fecha para almacenar la fecha de contratación del empleado.
- La clase CuentaBancaria tiene un atributo de tipo Cliente para almacenar la información del cliente propietario de la cuenta bancaria.
- La clase Escuela tiene un atributo de tipo Estudiante para almacenar la información referente a los alumnos de dicha escuela.

Ejemplo:

Un ejemplo de la relación de composición se encuentra en la clase Reloj, la cual tiene dos objetos de tipo Contador. A continuación, veremos a detalle la clase Contador:

```
① class Contador
  {
     private int numero;
     private int limite;          ②

③    public Contador
     ( int numero, int limite ) { this.numero = numero; this.limite = limte; }

     public int getNumero () {return numero;}
     public int getLimite () {return limite;}          ④

⑤    public void setNumero (int numero) { this.numero = numero}
     public void setLimite (int limite) { this.limite = limite}

     public void incrementa()
     {                            ⑥
        numero++;
        if ( numero > limite ) numero = 0;
     }

  }
```

Analicemos el código anterior, apoyándonos en los círculos rojos:

1. La clase es llamada Contador y tiene una visibilidad default parecida al public.
2. Esta clase contiene una variable global o atributo llamado número y límite; los dos tienen tipos de dato entero.
3. Tiene un constructor, el cual inicializa ambos atributos en un valor determinado.
4. Como los atributos de la clase son privados, debemos hacer uso de los métodos getters para obtener el valor de las variables.
5. Como los atributos de la clase son privados, debemos hacer uso de los métodos setters para asignar un valor de las variables.
6. La clase, además, contiene un método llamado incrementa, el cual cuenta de uno en uno hasta llegar a un número específico. Dicho método controla que el número no se pase del límite. Cuando el número alcanza el límite, lo regresa a cero (tal como sucede con los minutos de un reloj, los cuales inician en cero y, al llegar a 59, regresan a cero nuevamente).

Objetos que contienen arreglos

Hasta el momento, todos los ejemplos que hemos realizado son de clases cuyos atributos son datos simples como los enteros. Un objeto también puede contener un arreglo.

Ejemplo:

El siguiente programa muestra la clase Libro.
Un libro se caracteriza porque tiene un título, el año de edición y uno o más autores. Para manipular los autores, se incluyen en la clase dos atributos: el primero especifica la cantidad de autores que tiene el libro y el segundo un arreglo en el que se almacenarán los nombres de los autores.

```
class Libro
{
   private String titulo;
   private int edicion;
   private String [] autores;
   private int cantidadAutores;
   public Libro ( String titulo, int edicion,
                  String autores [], int cantidadAutores)
   {
    this.titulo = titulo;
    this.edicion = edicion;
    this.cantidadAutores = cantidadAutores;
    this.autores = new String[cantidadAutores];
    for(int k = 0; k < cantidadAutores; k++) this.autores[k] = autores[k];
   }
   void despliega()
   {
    System.out.print(autores[0]);
    for(int k = 1; k < cantidadAutores; k++)
         System.out.print(", " + autores[k]);
    System.out.println(", (" + edicion + "), " + titulo);
   }
```

Paquetes

¿Para qué se emplean los paquetes?

Los paquetes se emplean en Java para ayudar al programador a manejar la complejidad de los componentes de las aplicaciones que está desarrollando, ya que ayudan a agrupar clases que están relacionadas, de tal manera que se pueda prevenir el conflicto de nombres, controlar el acceso y localizar datos de forma rápida cuando se requiera.
Existen algunos paquetes que ya se encuentran definidos en Java, por ejemplo, el paquete "java.io", el cual hemos estado empleando porque contiene las clases requeridas para las funciones de entrada y salida de información (en efecto, a estos paquetes definidos en Java se les llama "librerías").
Nosotros podemos definir nuestros propios paquetes; de hecho, es una buena práctica de programación el agrupar las clases relacionadas que estamos implementando en paquetes que posteriormente podamos reutilizar para la elaboración de otras aplicaciones.

¿Qué se debe hacer para crear y emplear un paquete?

Para crear y emplear un paquete, debes realizar lo siguiente:

1. Define un nombre para el paquete.
2. En la primera línea del archivo, antes del nombre de la clase, escribe la instrucción package, seguido por el nombre que definiste.
3. Toma en cuenta que un archivo que contiene una clase solo puede estar dentro de un package.

```
package paqueteFechas;
public class Fecha {
     private int dia;
     private int mes;
     private int anio;
        }
```

Otros puntos para tomar en cuenta sobre los paquetes son los siguientes:

1. Un paquete puede contener varias clases.
2. Una clase puede referirse a otra del mismo paquete sin necesidad de agregar nada más.
3. Si una clase de un paquete quiere hacer uso de una clase de otro paquete, entonces se debe emplear la instrucción Import.