

## 01 - EJERCICIO DE PROGRAMACIÓN JAVA

### *Figuras de Superhéroes.*

*Práctica con clases básicas y composición de clases*

#### **Figuras de Superhéroes**

Se pide realizar una aplicación para una empresa que vende figuras de superhéroes. Para ello, tendrá que modelar todos los datos relativos a estas figuras. Así pues, se pide que programe las siguientes clases:

#### **Clase Superhéroe**

Esta clase definirá las características de un superhéroe.

#### Propiedades

Sus propiedades serán:

- Nombre (nombre del superhéroe)
- Descripción (cadena para describir brevemente su aspecto)
- Capa (booleano que indica si el superhéroe lleva o no capa)

#### Constructores

Haga un constructor con parámetros que reciba solo el nombre del superhéroe. La descripción se inicializará a vacío (cadena vacía) y la capa se inicializará al valor false (sin capa)

#### Métodos get y set

Programe los get y set para cada propiedad de la clase.

#### Método toString

Programe el método toString de forma que devuelva una cadena con todas sus propiedades.

## Clase Dimensión

Define un grupo de medidas de un objeto.

### Propiedades

Contiene las siguientes propiedades (todas ellas double):

- *Alto*. Medida correspondiente al alto del objeto.
- *Ancho*. Medida correspondiente al ancho del objeto.
- *Profundidad*. Medida correspondiente a la profundidad del objeto.

### Constructores

Haga un constructor sin parámetros que inicialice todas las propiedades a 0.

Haga un constructor que reciba como parámetro un alto, un ancho y una profundidad y asigne los valores a sus respectivas propiedades.

### Métodos get y set

Programa los get y set para cada propiedad.

### Métodos de cálculo

Programa un método llamado `getVolumen()` que devuelva el volumen máximo que ocuparía el objeto (alto x ancho x profundidad)

### Método toString

Programa el método `toString` de forma que devuelva una cadena con el alto, ancho, profundidad y volumen máximo del objeto.

## Clase Figura

Esta clase representará una figura de un superhéroe.

### Propiedades

- Código. Cada figura tiene un código identificativo formado por letras y números. Dos figuras distintas no tendrán el mismo código.
- Precio. Un double con el precio de la figura.

Además, puesto que la figura representa a un superhéroe, será necesario que la clase Figura contenga un objeto de la clase Superhéroe que describa las características de este.

Por otro lado, para definir las dimensiones de la figura, la clase Figura contendrá un objeto de la clase Dimensión

- Un objeto *dimensiones* de la clase Dimensión que defina las dimensiones del figura.
- Un objeto *superhéroe* de la clase Superhéroe que defina las características del superhéroe representado.

### Constructores

Programa un constructor que reciba como parámetro el código identificativo de la figura, su precio, un objeto Dimensión, un objeto Superhéroe. Estos valores se asignarán a cada propiedad.

### Métodos set y get

Programa un set para cada propiedad (código, precio, superhéroe y dimensiones). Igualmente un get para cada propiedad.

### Método toString

Programa el método toString de forma que devuelva una cadena con todas las características de la figura.

### Método modificador adicional

Programa un método `subirPrecio(double cantidad)` que reciba una cantidad de dinero. Este método subirá el precio actual de la figura en la cantidad indicada como parámetro.

### **Clase Colección**

La empresa vende a veces colecciones de figuras, que básicamente son conjuntos de figuras con una temática relacionada. Por ejemplo una colección “El Hombre Murciélago” que consta de una serie de figuras relacionadas con Batman, o una colección “Marvel”, con figuras de superhéroes de Marvel, etc...

### Propiedades

Las colecciones estarán descritas por la clase `Colección`, que tiene las siguientes propiedades:

- *nombreColeccion*, que será el nombre de la colección (“El Hombre Murciélago”, “Marvel”, etc.)
- Un objeto *listaFiguras*, que será un `ArrayList` que contendrá las distintas figuras que forman la colección.

### Constructor

Programa un constructor que reciba como parámetro el nombre de la colección. Este constructor construirá el `ArrayList` *listaFiguras*, (que estará vacío inicialmente)

### Metodos get y set

Programa un `get` y `set` para la propiedad *nombreColeccion*.

### Métodos modificadores

Programa un método añadirFigura(Figura fig), que reciba como parámetro una figura y la añada al listado de figuras de la colección.

Programa un método subirPrecio(double cantidad, String id) que reciba una cantidad de dinero y el identificador de una de las figuras de la colección. El método subirá el precio de dicha figura en la cantidad pasada como parámetro.

### Método toString y similares

Programa el método toString, de forma que devuelva una cadena con el listado de todas las figuras de la colección con todas las características de cada uno.

Programa un método conCapa() que devuelva una cadena con el listado de aquellas figuras de la colección que tengan capa.

### Otros métodos observadores

Programa un método masValioso() que devuelva la figura que tenga el precio mayor de la colección.

Programa un método getValorColeccion(), que devuelva un double con el precio total de la colección (es la suma de los precios de cada figura de la colección)

Programa un método getVolumenColeccion(), que devuelva un double con el volumen aproximado que ocuparía toda la colección de figuras. Para hacer este cálculo se deben sumar los volúmenes de cada figura, y añadirle al resultado el valor 200.

### **Programa de prueba**

Compruebe el funcionamiento de todas estas clases a través de un programa principal de prueba.