07. Ejercicios de clases (III)

Utiliza un paquete para implementar cada ejercicio (*com.acarballeira.ejercicio1*). Cada clases debe definirse en un archivo distinto. Crea las clases principales en un paquete denominado *drivers (com.acarballeira.drivers)*

1. Crear una clase para representar un ordenador.

   Desde el main asignar valores a sus campos y utilizar sus métodos.

2. Haz lo mismo que en el anterior ejercicio, pero con composición de objetos
3. Añadimos dos nuevas clases que heredan de Ordenador: Sobremesa y Portatil
   Un ordenador de Sobremesa tiene que tener entre sus componentes un Teclado y un Raton (nueva clases)
   Desde el main crear, utilizando constructores canónicos y pidiendo toda la información necesaria por teclado: 1 Sobremesa y 1 Portatil.
   Mostrar, utilizando toString(), el estado de ambos Ordenadores para comprobar el correcto funcionamiento del programa.
4. Igual que el ejercicio anterior, pero con todos los campos private.

   Desde el main asignar valores a sus campos y utilizar sus métodos.

5. Clase **Prueba** con los siguientes componentes:

   atributos (todos privados)

   private int[] lista; -> sin ningún tipo de acceso desde fuera de la clase
   private String nombre; -> dejamos que se consulte y/o cambie desde fuera
   private int factor; --> permitimos su consulta desde fuera

   métodos

   rellenarLista() -> rellena la lista con valores aletorios entre [0-factor)
   vaciarLista() -> pone todos los valores a cero
   devolverValorMasAlto() -> devuelve el valor más alto de la lista
   devolverValorMasBajo() -> devuleve el valor más bajo de la lista

   imprimirLista() -> imrprime la lista (formato libre)

   contiene -> le pasamos un valor y mira si está contenido en la lista, indicando en qué posición de la misma está la primera ocurrencia

   contiene2 -> le pasamos un valor y mira si está contenido en la lista, indicando en qué posición de la misma está la última ocurrencia

   devolverValoresMayoyYMenor -> devuelve los valores mayor y menor de la lista

6. Programa compuesto por dos ficheros '.java'. En uno de ellos definiremos la clase Bicicleta la cual estará formada por:

   Atributos: (todos privados)

   presion array de dos enteros, indicando las presiones de las ruedas delantera y trasera respectivamente.
   color entero indicando el color de la bicicleta.
   tipo carácter o string indicando el tipo de bicicleta.
   kmsCadena entero indicando el número de kilómetros recorridos con la cadena actual.

   Métodos:

setter/getter para cada atributo.

Para los métodos relativos a la presión, habrá dos versiones: una que trate ambas ruedas en conjunto y otra que tome como parámetro el número de rueda a tratar.

comprobarEstadoCadena imprimirá un texto en función de los kms recorridos:

<= 1800 La cadena está en buen estado.
>1800 && <=2000 La cadena está demasiado gastada.
>2000 SUSTITUIR INMEDIATAMENTE LA CADENA.

PROGRAMA PRINCIPAL:

a) Crear una bicicleta.

b) Fijar cada uno de sus atributos.

El color se escogerá aleatoriamente un valor entre 1-10

c) Imprimir su estado (presiones, color, tipo, kms y estado de la cadena).

7. Desde el ejercicio anterior.
   Nuevas clases: BicicletaCarretera y BTT que hereden de Bicicleta.

   BicicletaCarretera añade el atributo dropManillar

   BTT añade los atributos recorridoSuspensionDelantera, recorridoSuspensionTrasera

   El método de Bicicleta, comprobarEstadoCadena, dejará de imprimir y en su lugar devolverá un String

   En BicicletaCarretera se debe hacer un override de este último método, de modo que el primer aviso sea a los 5000km y el segundo a los 7000km

   Crear constructores en todas las clases.

8. Programa para la gestión de **figuras geométricas**.

   1) Deben existir las siguientes clases:

   - Punto
   - Cuadrilatero
   - Cuadrado
   - Rectangulo
   - Circulo
   - Triangulo

   2) Todos las figuras (el punto **NO** es una figura) deben permitir:

   - pintarlas de un color: valores válidos 1-10
   - girarlas
   - desplazarlas
   - redimensionarlas
   - calcular su área
   - calcular su perímetro
   - mostrarlas
   - ocultarlas
   - Imprimir su estado, poniendo además el color como nombre (la relación entre número y nombre es decisión del programador).

   Utilizar una **interfaz** para asegurarse de que cualquier figura proporciones los métodos indicados en el punto 2)

Esta interface debe además proporcionar los **nombre de los colore**s validos (entre el 1 y el 10)