1. **La clase Coche contiene los atributos:**

marca, modelo, precio base, descuento y una lista máxima de 3 accesorios (generar un

array). El descuento es una cantidad que se aplica a todos los coches por igual.

1. **Los accesorios solo pueden ser de tres tipos, Techo, Asientos calefactables,**

**Navegador.** Cada accesorio tiene un precio prefijado que puede cambiar si el

proveedor lo decide. Generar una clase Accesorio interna a Coche y utilizar datos

enumerados.

1. **Definir los métodos siguientes:**

* arrancar, detenerse, acelerar, frenar: tened en cuenta que un coche no puede acelerar si no está arrancado, no puede detenerse si no ha frenado previamente y no puede frenar si no ha acelerado.
* verPrecio: devolverá el precio final (puede tener decimales). Tendrá en cuenta el precio base, el descuento y los accesorios contratados,

**package** segunda\_practica\_evaluable;

**public** **class** coche {

//ATRIBUTOS

**private** String marca;

**private** String modelo;

**private** **double** precioBase;

**private** **static** **double** *descuento*;

//array 3 accesorios

**private** Accesorio[] accesorios = **new** Accesorio[3];

//CONSTRUCTOR

**public** coche(String marca, String modelo, **double** precioBase) {

**this**.marca = marca;

**this**.modelo = modelo;

**this**.precioBase = precioBase;

}

//GETTER Y SETTER

**public** **static** **void** setDescuento(**double** descuento) {

coche.*descuento* = descuento;

}

**public** **void** addAccesorio(Accesorio accesorio) {

**for** (**int** i = 0; i < accesorios.length; i++) {

**if** (accesorios[i] == **null**) {

accesorios[i] = accesorio;

**break**;

}

}

}

//MÉTODOS

**private** **boolean** arrancado = **false**;

**private** **boolean** frenado = **false**;

**private** **boolean** acelerado = **false**;

//ARRANCAR, DETENERSE, ACELERAR, FRENAR, VERPRECIO

**public** **void** arrancar() {

arrancado = **true**;

}

**public** **void** detenerse() {

**if** (frenado) {

arrancado = **false**;

}

}

**public** **void** acelerar() {

**if** (arrancado) {

acelerado = **true**;

}

}

**public** **void** frenar() {

**if** (acelerado) {

frenado = **true**;

}

}

**public** **double** verPrecio() {

**double** precioFinal = precioBase;

**for** (Accesorio accesorio : accesorios) {

**if** (accesorio != **null**) {

precioFinal += accesorio.getPrecio();

}

}

precioFinal -= precioFinal \* *descuento*;

**return** precioFinal;

}

//CLASE ACCESORIOS

**private** **class** Accesorio {

**private** String tipo;

**private** **double** precio;

/\*public void accesorio(Tipo tipo, double precio) {

this.tipo = tipo;

this.precio = precio;

}\*/

//GETTER Y SETTER

**public** **double** getPrecio() {

**return** precio;

}

**public** String getTipo() {

**return** tipo;

}

**public** **void** setTipo(String tipo) {

**this**.tipo = tipo;

}

}

**public** **enum** Tipo {

***TECHO***, ***ASIENTOS\_CALEFACTABLES***, ***NAVEGADOR***

}

}

1. **Generar nuevas clases que hereden de esa clase con atributos específicos.**

* **CocheElectrico**
* Batería que tiene dos atributos: potencia de la batería y consumo (el consumo será predefinido en función de la potencia).

Los coches eléctricos podrían tener un descuento distinto al que se aplica a los coches.

(no es acumulable con respecto al que pudieran tener los coches en general). Este

descuento aplicará por igual a todos los coches eléctricos que se den de alta.

**package** segunda\_practica\_evaluable;

**public** **class** CocheElectrico **extends** coche{

**public** **int** bateria;

**public** String consumo;

//constructor

**public** CocheElectrico(String marca, String modelo, **double** precioBase) {

**super**(marca, modelo, precioBase);

}

//getter y setter

**public** **int** getBateria() {

**return** bateria;

}

**public** **void** setBateria(**int** bateria) {

**this**.bateria = bateria;

}

**public** String getConsumo() {

**return** consumo;

}

**public** **void** setConsumo(String consumo) {

**this**.consumo = consumo;

}

}

* **CocheCombustible**
* Deposito que tiene dos atributos: tipo que puede ser diésel o gasolina, etiqueta: puede ser B (cuando el depósito es de gasolina) o C (cuando el depósito es diésel)

En los tipos de coche sobrescribir el método de la clase padre verPrecio dándole un

comportamiento distinto: si es eléctrico se aplica un descuento del 10% sobre el

precio base y si es de combustible, se incrementa un 10% si es diésel.

**package** segunda\_practica\_evaluable;

**public** **class** CocheCombustible **extends** coche{

**public** String tipo;

**public** String etiqueta;

//constructor

**public** CocheCombustible(String marca, String modelo, **double** precioBase) {

**super**(marca, modelo, precioBase);

}

//getter y setter

**public** String getTipo() {

**return** tipo;

}

**public** **void** setTipo(String tipo) {

**this**.tipo = tipo;

}

**public** String getEtiqueta() {

**return** etiqueta;

}

**public** **void** setEtiqueta(String etiqueta) {

**this**.etiqueta = etiqueta;

}

}

1. **Realice un programa principal que utilice los vehículos.**

* Generando varios objetos de tipo Coche y usa sus métodos.
* Generar un array con elementos de tipo coche. Después mostrar todos los coches del array ordenados por precio.
* Crear un array de coches que mezcle coches, coches eléctricos y coches de combustible y utilizar los métodos de acción arrancar, detenerse,...
* Recorrer el array anterior de coches mezclados y mostrar el precio total de los coches introducidos.
* Generar un método que me muestre un informe del total de coches de una marca dada por parámetro (el método solo tendrá este parámetro). Mostrar un contador de coches total de esa marca, desglosando el total de eléctricos, el total de combustible y el resto que no sean ni eléctricos ni de combustible

**package** segunda\_practica\_evaluable;

**public** **class** Vehiculos {

**public** **static** **void** main(String[] args) {

}

**public** **void** informeTotal() {

}

}