Gestión de Vehículos		
1º DAW - ETS		
Departamento de Informática	**Fecha**	**Luke Eric Marten Llorente**

Descripción

Se desarrollará un sistema de gestión de vehículos mediante una jerarquía de clases. La clase base `Vehículo` definirá los atributos y métodos comunes a todos los vehículos. A partir de esta, se implementarán clases derivadas para `Coche`, `Moto` y `Camión`, cada una con atributos y comportamientos específicos.

Requisitos

Clase `Vehículo` (Base)

Propiedades

- `String marca`: Marca del vehículo.
- `String modelo`: Modelo del vehículo.
- `int añoFabricacion`: Año de fabricación.
- `double precio`: Precio del vehículo.

Métodos

- **Constructores**:
  - Uno que inicializa todos los atributos.
  - Otro que inicializa `marca`, `modelo`, `añoFabricacion` y asigna `0` al precio.
- **Getters y Setters** para todos los atributos.
- **`toString()`**: Devuelve una cadena con la información en el formato:
  - `"Marca: [marca] - Modelo: [modelo] - Año: [añoFabricacion] - Precio: [precio]"`.
- **Método abstracto `calcularMantenimiento()`**: Cada subclase deberá implementarlo.

Clase `Coche` (Derivada de `Vehículo`)

Propiedades

- `int numeroPuertas`: Número de puertas.
- `boolean esElectrico`: Indica si el coche es eléctrico.

Métodos

- **Constructores**:
  - Uno que inicializa todos los atributos.
  - Otro que inicializa `marca`, `modelo`, `añoFabricacion`, `numeroPuertas` y `esElectrico`, asignando `0` al precio.
- **`calcularMantenimiento()`**:
  - Si el coche es eléctrico → **200€**
  - Si no lo es → **500€**
- **`toString()`**: Devuelve una cadena con la información adicional:
  - `"Marca: [marca] - Modelo: [modelo] - Año: [añoFabricacion] - Precio: [precio] - Puertas: [numeroPuertas] - Eléctrico: [esElectrico]"`.

Clase `Moto` (Derivada de `Vehículo`)

Propiedades

- `int cilindrada`: Cilindrada del motor.
- `boolean tieneSidecar`: Indica si la moto tiene sidecar.

Métodos

- **Constructores**:
  - Uno que inicializa todos los atributos.
  - Otro que inicializa `marca`, `modelo`, `añoFabricacion`, `cilindrada` y `tieneSidecar`, asignando `0` al precio.
- **`calcularMantenimiento()`**:
  - Con sidecar → **300€**
  - Sin sidecar → **150€**
- **`toString()`**: Devuelve una cadena con la información adicional:
  - `"Marca: [marca] - Modelo: [modelo] - Año: [añoFabricacion] - Precio: [precio] - Cilindrada: [cilindrada] - Sidecar: [tieneSidecar]"`.

Clase `Camion` (Derivada de `Vehículo`)

Propiedades

- `double capacidadCarga`: Capacidad de carga en toneladas.
- `int numeroEjes`: Número de ejes del camión.

Métodos

- **Constructores**:
  - Uno que inicializa todos los atributos.
  - Otro que inicializa `marca`, `modelo`, `añoFabricacion`, `capacidadCarga` y `numeroEjes`, asignando `0` al precio.
- **`calcularMantenimiento()`**:
  - **Costo anual:** `300€ por cada eje + (50€ por tonelada de capacidad de carga)`.
- **`toString()`**: Devuelve una cadena con la información adicional:
  - `"Marca: [marca] - Modelo: [modelo] - Año: [añoFabricacion] - Precio: [precio] - Carga: [capacidadCarga]T - Ejes: [numeroEjes]"`.

Clase `Main_Vehiculos`

- Crear instancias de diferentes tipos de vehículos (`Coche`, `Moto` y `Camión`).
- Almacenar los vehículos en un array de tipo `Vehículo`.
- Recorrer el array e imprimir la información de cada vehículo con `toString()`.
- Calcular y mostrar el costo anual de mantenimiento con `calcularMantenimiento()`.