Módulo de PROGRAMACIÓN. SEGUNDO TRIMESTRE. ***PRÁCTICO.***
Nombre y dos apellidos: _____
Interpreta las siguientes clases, interfaces, enumerados y otras definiciones que se describen a continuación. Codifica las propiedades descritas (privadas), sus constructores, getters (todos), setters y otros métodos que se indiquen. Puedes cambiar el tipo de dato de las propiedades, excepto TipoCombustible, si lo crees conveniente.
Primera parte. Escribe el código java de:
El tipo enumerado **Combustible** con los DIESEL, GASOLINA, HIBRIDO, ELECTRICO y AGUA.
La clase **Motor** tiene una potencia (medida en caballos), un número de cilindros y un tipo de combustible.
public Motor (int potencia, int cilindros, Combustible combustible) { }
 Comprueba que los parámetros son válidos y lanza una excepción de usuario (dos niveles) cuando sean incorrectos: menores que cero, nulos, tipo de combustible erróneo …

La interface **Movible**, derivada de Comparable. Describe:
 Un array con el impuesto que se aplica al tipo de combustible. Por ejemplo:

[10.0, 5.5, 3.2, 1.0, 0.0].
 Métodos consultores para todas las propiedades de la clase Vehículo y modificador sólo del precio: más abajo se describen las propiedades.
 Método que retorna un doublé con el valor de los impuestos del vehículo. Se verá cómo se calculan.

La clase Vehículo tiene una matrícula, un precio y un motor.
public Vehiculo (String matricula, int precio, Motor motor) { }
 Implementa Movible. Codifica los métodos que se puedan.
 Comprueba que los parámetros son válidos y lanza una excepción de usuario (dos niveles) cuando sean incorrectos: menores que cero, nulos …
 El orden natural de los vehículos es por matrícula y a igualdad de ésta, aunque no suceda, por potencia.

Las siguientes clases derivadas de la clase Vehículo:
 Coche que tiene un número de plazas.

public Coche (String matricula, int precio, Motor motor, int numPlazas) { }
public Coche (String matricula, int precio, int potencia, int cilindros, Combustible combustible, int numPlazas) { }
public Coche (String coche) { } // cadena con las propiedades separadas por comas.

- Los impuestos se calculan incrementando el precio en el porcentaje correspondiente según su combustible y en el porcentaje correspondiente al número de plazas. Por ejemplo: 5% para cinco plazas).
- El orden natural de los coches es el del vehículo, y a igualdad de éste por número de plazas.

 Camión que tiene un peso y puede tener o no un remolque.

public Camion (String matricula, int precio, Motor motor, int peso, Remolque remolque) { }

- **Remolque** es una clase que tiene una carga máxima a transportar y un peso, medidos en toneladas.

public Remolque (int cargaMaxima, int peso) { }

- Debemos permitir añadir un **Remolque** al **Camión** o cambiar el que tiene por otro.
- El **Camión** permite conocer su carga máxima que es la misma que tiene su Remolque; si o tiene remolque será cero.
- El **Camión** permite conocer su peso, que es el suyo y el del Remolque juntos, si lo tiene.
- Los impuestos se calculan incrementando el precio en el porcentaje correspondiente según su combustible, en un 0.2% cada 100 caballos de potencia y en un 0.1% cada 1000 kg de carga máxima permitida, si tiene remolque claro.

Segunda parte.

La empresa "TeLoVendemos" dispone para la venta de una flota de vehículos almacenados en una nave de miles de metros cuadrados. Realiza las tareas que se describen a continuación precediendo cada método con la descripción de la tarea que realiza.

1. La empresa **TeLoVendemos** dispone de:
   - Una *nave* para aparcar **vehículos,** de todos los tipos: es una estructura de datos del tipo que elijas.
   - Una *capacidad* máxima: número máximo de vehículos que caben de la nave.
2. Describe el constructor por defecto.
3. Aparca nuevos vehículos en la nave. Si no caben, lanza la RunTimeException más adecuada para la situación. El nombre del método es el mismo para todos:

public void aparca (lista de parámetros)

- Un vehículo en la nave.
- Todos los vehículos procedentes de otra nave de vehículos.
- Todos los vehículos procedentes de la empresa "TeLoCompramos". Se duplica la capacidad de la nave.

4. Método que retorna en una nueva nave todos los camiones, ordenados por potencia, que tienen el combustible pasado como parámetro y que no tiene remolque asociado.
5. Elimina de la nave el vehículo que se corresponda con una matrícula dada.
6. Vende de la nave todos los vehículos cuyo combustible (*como una cadena*) sea el proporcionado como parámetro. Lanza una excepción si no existe el combustible en el tipo enumerado.

public void elimina (String combustible)

7. ¿Cuál es el vehículo de mayor precio? Usa un protecio de comparación alternativo.
8. ¿Cuál es el vehículo DIESEL de mayor precio?

Tercera parte.

Crea una clase para probar clases y tareas que has realizado. Captura todas las excepciones que se produzcan.

1. Instancia tres o más motores con distinto combustible.
2. Instancia tantos coches como constructores tenga la clase. Usa distintos motores.
3. Instancia varios camiones: con y sin remolque y distintos motores.
4. Crea una empresa.
5. Aparca algún vehículo en la nave de la empresa.
6. Añade a una nave un grupo de vehículos y apárcalos en la nave de la empresa.
7. Otras.