Parcial 1 - Algoritmos I Taller: Tema D

Debés entregar el código completo en los campos correspondientes de cada ejercicio del formulario de entrega y subir el archivo .hs correspondiente. Este código debe poder ejecutarse en haskell sin errores. Te recomendamos para ello que pruebes con diferentes ejemplos antes de entregar.

Descripción del problema:

En un sistema de gestión de automotores representaremos **autos** y **camiones**, con información acerca de su motor, su nivel de combustible actual y atributos adicionales según el tipo de vehículo.

Ejercicio 1:

Definir el tipo **Automotor** que consta de dos constructores `Auto` y `Camion` con los siguientes parámetros:

4. El constructor **Auto** debe tomar como parámetro: el tipo de motor (que puede ser únicamente una de las opciones Nafta, Diesel, Electrico o Hibrido), la cantidad de gasolina en litros y una lista de accesorios (cada uno puede ser cualquier string, por ejemplo "Aire Acondicionado").

5. El constructor **Camion** debe tomar como parámetro: el tipo de motor (mismas opciones que el constructor Auto), la cantidad de gasolina en litros, la cantidad de ruedas y el tonelaje (un entero indicando cuánto pesa)

Ejercicio 2:

A partir del tipo definido en el punto anterior, definí los siguientes **términos**:

```
sedanFull :: Automotor
sedanFull = <COMPLETAR>
```

que debe representar un auto con motor a nafta, 40 litros de gasolina y accesorios Aire Acondicionado, Ventanas eléctricas y ABS.

```
cityEV :: Automotor
cityEV = <COMPLETAR>
```

que debe representar un auto con motor eléctrico, 0 litros de gasolina y el único accesorio pantalla táctil.

```
camionMinero :: Automotor
camionMinero = <COMPLETAR>
```

que debe representar un camión con motor diesel, 120 litros de gasolina, 10 ruedas y 35 toneladas de tonelaje.

```
camionMudanza :: Automotor
camionMudanza = <COMPLETAR>
```

que debe representar un camión pequeño de mudanzas, con motor Diesel, 60 litros de gasolina, 4 ruedas y 3 toneladas de tonelaje.

Ejercicio 3:

Definir las siguientes funciones:

4. `esMotorElectrico :: Automotor -> Bool`

   Devuelve `True` si el automotor es un auto cuyo *motor* es eléctrico o híbrido.
   Devuelve `False` en cualquier otro caso.

5. `camionPesadoDiesel :: Automotor -> Bool`
   Devuelve `True` si el automotor es un camión con motor diesel y cuyo tonelaje es mayor a 20. Devuelve `False` en cualquier otro caso.

Ejercicio 4:

Definir la función

```
contarTanquesVacios :: [Automotor] -> Int
```

que, dada una lista de automotores, devuelve la cantidad de vehículos cuya tanque de gasolina está vacío.

Ejercicio 5:

Definir el tipo recursivo **cola de automotores**:

```
data ColaAutomotor = Vacia | Encolada Automotor ColaAutomotor
  deriving Show
```

A partir de este tipo, definir la función

```
autosConAlMenosDosAccesorios :: ColaAutomotor -> ColaAutomotor
```

que, dada una cola de automotores q, devuelve la cola que contiene únicamente los autos que poseen al menos dos accesorios, en el mismo orden en que aparecen en q.

AYUDA: puede usar la función `length` que dada una lista devuelve la cantidad de elementos que tiene.