Parcial 1 - Algoritmos I Taller: Tema E

> *Debés entregar el código completo en los campos correspondientes de cada ejercicio del formulario en el que completaste tus datos personales. 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 este examen se plantea la necesidad de representar trenes de carga, en el que los vagones pueden ser de diferentes tipos, como vagones de granos (pueden transportar trigo, soja, arroz, etc) y vagones cisterna (que transportan líquidos como petroleo, agua, oxigeno liquido, etc). Cada tipo de vagón tiene características específicas que se tiene en cuenta en sus constructores.

Ejercicio 1

a)

Definir el tipo de dato `Vagon` que consta de dos constructores, `Granos` y `Cisterna` que tienen los siguiente parámetros:

- El constructor `Granos`, toma como parámetros el nombre del grano que carga, el número de ejes del vagón y las cantidad de toneladas que transporta.
- El constructor `Cisterna`, que toma como parámetros el líquido que transporta, la cantidad de litros transportados (medido en miles de litros), la temperatura de operación en grados centígrados (la temperatura a la que se mantiene el vagón).

b)

A partir del tipo definido en el punto anterior, definir las siguientes constantes:

- ```
  vagonPetroleo :: Vagon
  vagonPetroleo = <COMPLETAR>
  ```

  que se corresponde a un vagón cisterna que transporta `"Petroleo"`, en cantidad de `30` mil litros y que tiene una temperatura de operación de `18.5` grados.

- ```
  vagonTrigo :: Vagon
  vagonTrigo = <COMPLETAR>
  ```

que se corresponde a un vagón de granos que transporta `"Trigo"`, donde el vagón tiene `2` ejes y transporta una cantidad de `40` toneladas.

c)

Definir la función `excesoDeGranos :: Vagon -> Bool` que dado un vagón, verifica si el vagón de granos transporta más de 120 toneladas de granos.

d)

Definir la función `cisternaLleno :: Vagon -> Int -> Bool` que dado un vagón y un valor máximo de miles de litros `k`, verifica si un vagón cisterna está lleno, esto será así si el vagón es cisterna y transporta una cantidad mayor o igual a `k`, en cualquier otro caso debe devolverse `False`.

e)

Definir la función `mayorCantidadEjes :: [Vagon] -> Int` que dada una lista `xs` de vagones, debe devolver la cantidad de ejes que tiene el vagón de granos con más ejes en de la lista `xs`.

Ejercicio 2

Dado el tipo recursivo `Tren` definido de la siguiente manera

```
data Tren = SinVagones | Encadena Vagon Tren
```

podemos definir `ejemploTren` de la siguiente manera

```
ejemploTren :: Tren
ejemploTren = Encadena vagonPetroleo (Encadena vagonTrigo SinVagones)
```

Definir la función `soloCisternaTemp :: Tren -> Float -> Tren` que dado un tren `ts` y una temperatura máxima `maxTemp` devuelve un nuevo tren con sólo los vagones de `ts` que son cisterna y tienen una temperatura de operación menor o igual a `maxTemp`.