FUNDAMENTOS DE COMPUTACIÓN
TRABAJO ENTREGABLE 5
MAYO 2022

Este trabajo tiene un puntaje de 5 puntos, y debe ser realizado en forma **INDIVIDUAL**.
Se debe **subir por Aulas** antes del 29/5 a las 21hs.

Conjuntos como Listas

Se desea representar conjuntos como listas de elementos sin repeticiones.
Para ello, definimos el tipo de los conjuntos de elementos de tipo `a` mediante la siguiente declaración de tipos en Haskell:

```
type Conj a = [a]
```

Por ejemplo, el conjunto que contiene los primeros 5 primos será representado por la siguiente lista, u otra que tenga los mismos elementos pero en cualquier otro orden:

```
prim :: Conj Int
prim = [2,3,5,7,11]
```

Se pide definir las siguientes funciones, utilizando como funciones auxiliares sólo la igualdad `(==)` entre elementos de las listas, los conectivos booleanos `(&&),(||), not`, y las funciones definidas en este mismo trabajo:

a) `pert :: Eq a => a -> Conj a -> Bool`, que determina si un elemento pertenece a un conjunto.
Ejemplos: `pert 3 [4,1,3,2] = True`
`pert 'f' "abcde" = False`

b) `esConj :: Eq a => [a] -> Bool`, que determine si una lista representa a un conjunto (o sea, no contiene elementos repetidos).
Ejemplos: `esConj [4,1,3,2] = True`
`esConj "abcbaede" = False`

c) `inter :: Eq a => Conj a -> Conj a -> Conj a`, que calcula la intersección de dos conjuntos.
Ejemplos: `inter [1,2,3] [4,5,6] = []`
`inter "facgbh" "abcde" = "acb"`, en este o cualquier otro orden

d) `union :: Eq a => Conj a -> Conj a -> Conj a`, que calcula la unión de dos conjuntos.
Ejemplo: `union [1,2,3,4] [2,4,6] = [1,2,3,4,6]`, en este o cualquier otro orden

e) `incl :: Eq a => Conj a -> Conj a -> Bool`, que dados dos conjuntos determina si el primero está incluido en el segundo, sin importar el orden de los elementos.
Ejemplos: `incl [1,2,3] [3,4,2,1,5] = True`
`incl "abc" "edfbcw" = False`

f) `iguales :: Eq a => Conj a -> Conj a -> Bool`, que determina si dos conjuntos son iguales (esto es, si tienen los mismos elementos, sin importar el orden de aparición).
Ejemplos: `iguales [1,2,3] [3,2,1] = True`
`iguales "abc" "cb" = False`

Importante: en las funciones c), d), e) y f) que reciben listas que representan conjuntos se puede asumir que éstas *no contienen elementos repetidos*, y el resultado también debe ser una lista sin elementos repetidos.

ENTREGA:

- Se deberá subir un único archivo Haskell (`.hs`) con el código fuente de la solución. En Aulas se encuentra el archivo `Entregable5.hs` con las funciones que deben implementarse y algunos conjuntos para hacer pruebas. Solicitamos utilizarlo como template para facilitar la corrección.

- **No se corregirán archivos que no compilen**, por lo que recomendamos comentar el código que no compile y dejar como `undefined` las funciones no implementadas.