LABORATORIO 2022 – Programación III
Programación Funcional: Lenguaje Haskell

Usar el código del **Apéndice A** para codificar el laboratorio.
El archivo con el laboratorio a entregar debe tener como nombre:

- ***LabH2022PIII<Apellido del alumno>*.hs**
  ejemplo: LabH2022PIII_Perez.hs

El laboratorio debe ser presentado de dos formas:

1. por mail a la cuenta: **progUNSLCMN@gmail.com**. En tema del mail se deberá colocar el nombre del archivo sin la extensión.
   ejemplo: LabH2022PIII_Perez
2. subido en el aula Virtual de la materia, en la ***tarea habilitada*** en la pestaña P.Funcional/Práctico/Lab

Fecha entrega: hasta el jueves 8/9– 24hs
IMPORTANTE !! El Laboratorio tiene que estar entregado para poder rendir la recuperación del 1er. parcial de Haskell del viernes 9/9

Ejercicio 1: implementar la función de orden superior ***mapTup***, que se aplica sobre listas de tuplas de dos elementos de tipo:

```
mapTup :: (a -> b) -> (c -> d)-> [(a,c)] -> [(b,d)]
```

Esta función toma como argumentos dos funciones y una lista de tuplas de dos elementos y retorna una lista de tuplas de igual longitud a la lista de entrada, donde cada tupla fue transformada en otra aplicando la función entrada como 1er arg. al 1er. elemento y la función entrada como 2do argumento, al 2do elemento de cada tupla respectivamente.
Ejemplo de uso:

```
*LabHask2022> mapTup (+1) (=='a') [(20,'a'),(3, 'b'),(54,'a')]
*LabHask2022> [ (21,True),(4,False),(55, True)]
```

Definir tres variantes:

a) una función recursiva (***mapTupR***) que refleje su forma de razonar la solución. Indicar en comentario en código que tipo de recursión implementó (de cola o controlada).
b) una función recursiva con acumuladores (***mapTupRA).*** En caso de no hacerla explique porqué en comentario.
c) dos funciones, usando respectivamente las funciones predefinidas de orden superior foldl (***mapTupfl***) y foldr (***mapTupfr***) . Usar una definición Lambda en los folds. En caso de no hacer alguno o ambos casos, justifique en comentario.

Ejercicio 2: implementar la función de orden superior ***filterTup,*** que se aplica sobre listas de tuplas de dos elementos de tipo:

```
filterTup :: (a -> Bool) -> (b -> Bool) -> [(a,b)] -> [(a,b)]
```

Esta función toma como argumentos dos predicados y una lista de tuplas de dos elementos y retorna una lista de igual o menor longitud que la lista entrada. La lista resultado contiene aquellas tuplas que satisfacen los dos predicados pasados como 1ero. y 2do argumentos en sus 1ro y 2do. elementos.

Ejemplo de uso:

```
*LabHask2022> filterTup (>1) (=='a') [(20,'a'),(3,'b'),(54,'a')]
*LabHask2022> [(20,'a'),(54,'a')]
*LabHask2022> filterTup (>1) (=='a') [(20,'c') ,(1,'a'),(54,'b')]
*LabHask2022> [ ]
```

Definir tres variantes:

a) una función recursiva (***filterTupR)*** que refleje su forma de razonar la solución. Indicar en comentario en código que tipo de recursión implementó (de cola o controlada).

b) una función recursiva con acumuladores (***filterTupRA***). En caso de no hacerla explique porqué en comentario.

c) dos funciones, usando respectivamente las funciones predefinidas de orden superior foldl (***filteTuprfl***) y foldr (***filterTupfr***). Usar definiciones locales parea 1er. arg. de los folds. En caso de no hacer alguno o ambos casos, justifique en comentario.

Ejercicio 3: implementar la función de orden superior **takeTup**, de tipo:

```
takeTup:: Int -> [(a,b)] -> [(a,b)]
```

Esta función toma como argumentos un entero n y una lista de tuplas de dos elementos y retorna una lista de tuplas con los n primeros elementos del 2do. argumento.

Ejemplo de uso:

```
*LabHask2022> takeTup 2 [(20, 'a'),(3,'b'),(54,'a')]
*LabHask2022> [(20, 'a'),(3,'b')]
*LabHask2022> takeTup 5 [(20,'a'),(3,'a'),(54, 'a')]
*LabHask2022> [(20,'a'),(3,'a'),(54, 'a')]
```

Definir tres variantes:

a) una función recursiva (***allTupR***). que refleje su forma de razonar la solución. Indicar en comentario en código que tipo de recursión implementó (de cola o controlada)

b) una función recursiva con acumuladores (***allTupRA***). En caso de no hacerla explique porqué en comentario.

c) dos funciones, usando las funciones predefinidas de orden superior foldl (***allTupfl***) y foldr (***allTupfr***), respectivamente. Usar funciones anónimas en los folds (expresiones Lambda). En caso de no hacer alguno o ambos casos, justifique en comentario.

Ejercicio 4 :

- *Leer* la Guía de Haskell, punto 7.1 (y extender tema "sinónimos de tipo" con lectura en libro).

- *Analizar* el modelo simplificado de un *Sistema de Stock para un comercio* (usando sinónimos de tipos) que se provee en el código proporcionado en el **Apéndice A**

- *Ejecutar* en el interprete las siguientes expresiones y analizar los resultados.

```
*LabHask2022> tabla1S
*LabHask2022> map item tabla1S
*LabHask2022> filter ((==200).codItem)tabla1S
```

- *Leer* **Apéndice B.**

a) Defina la función **listaItems** que toma como argumento una relación (tabla) de Stock y retorna como resultado una nueva relación con cuatro atributos: codItem, ítem, marca y rubro. Esta operación correspondería a la operación $\pi_{\text{CodItem, Item,Marca,Rubro}}(\text{tabla1S})$ (ver **Apéndice B**)

Ejemplo:

```
*LabHask2022> listaItems tabla1S
```

```
[(100,"ARROZ GRANO GRANDE","CONDOR","Alimentos"),(107,"ARROZ
GRANO GRANDE","GALLO","Alimentos"),(200,"ACEITE DE
GIRASOL","NATURA","Alimentos"),(200,"ACEITE DE
GIRASOL","COCINERO","Alimentos"),(410,"AGUA MINERAL S/GAS BAJO
SODIO","SER","Alimentos"),(412,"AGUA SABORIZADA LIMA
LIMON","SER","Alimentos"),(478,"ALFAJOR CHOCOLATE
TITA","TERRABUSI","Alimentos"),(479,"ALFAJOR CHOCOLATE
RODESIA","TERRABUSI","Alimentos"),(708,"LECHE
DESC.PASTEURIZADA","SERENISIMA","Alimentos"),(767,"ARVEJAS SECAS
REMOJADAS","NOEL","Alimentos"),(801,"ANTITRANSPIRANTE ROLL-
ON","ETIQUET","PERFUMERIA")]
```

b) Defina la función **itemsReponer** que toma como argumento una tabla de Stock y retorna como resultado una relación (tabla) de Stock y retorna como resultado una nueva relación donde se encuentran solo los ítems de stock (con todos sus atributos) del almacén cuya existencia en depósito se encuentre por debajo del valor mínimo recomendado para dicho producto. . Esta operación correspondería a la operación $\sigma_{CantExis<=VMin}$ **(tabla1S)** (ver **Apéndice B**)

Ejemplo:

```
*LabHask2019> itemsReponer tabla1S
[(410,"AGUA MINERAL S/GAS BAJO
SODIO","SER","Alimentos",31,"1.5LT",20,50,3000,10.0,35),(479,"AL
FAJOR CHOCOLATE
RODESIA","TERRABUSI","Alimentos",31,"40GR",9,200,3500,4.0,30),(8
01,"ANTITRANSPIRANTE ROLL-
ON","ETIQUET","PERFUMERIA",20,"60gr",30,45,2000,25.0,30)]
```

Apéndice A

```
-------------------------------------------------
-- Module : LabH2C2022
-- Developer : Apellido y Nombre del alumno
--
-- Programacion III - Laboratorio Haskell 2022
-------------------------------------------------
module LabH2C2022 where

-- Funciones solicitadas en ejs.1,2,3

-- Funcion solicitada en ej.4.a) listaItems

-- Funcion solicitada en ej.4.b) itemsReponer

-------------------------------------------------
-- MODELO SISTEMA DE STOCK DE ALMACEN
-------------------------------------------------
type CodItem = Int    -- Codigo de cada  producto en la BD
type Item    = String -- Descripcion del producto
type Marca   = String
type Rubro   = String
type UMed    = String --Unidad de Medida:1LT,800GRM, 1500CM3,etc
type CantExis=Int     --cantidad de productos en deposito
type VMin    = Int    -- valor en existencia recomendado para
                      -- solicitar reposicion
type VMax    = Int    -- valor maximo de acopio en deposito
type PrecioU = Float  -- precio de compra por unidad de medida
type PGan    = Int    -- Porcentaje de de ganancia a aplicar
                      -- sobre el precio de compra
type CodProv = Int       -- Codigo de cada proveedor en la BD
type NombreP = String    -- Nombre del proveedor en la BD
type DireccionP = String
type CelularP = String

-- tupla con datos de un item de Stock
type ItemStock = (
  CodItem,
  Item,
  Marca,
  Rubro,
  CodProv,
  UMed,
  CantExis,
  VMin,
  VMax,
  PrecioU,
```

```
  PGan
 )
-- tupla con datos de 1 proveedor
type Proveedor = (
  CodProv,
  NombreP,
  DireccionP,
  CelularP
 )

-- Tablas BD
type TStock = [ItemStock] --Tabla con el Stock de un comercio
type TProveedor=[Proveedor] --Tabla con proveedores del comercio

--funciones de extracción
codItem :: ItemStock -> CodItem
codItem
(cod,item,marca,rubro,prov,umed,cant,cmin,cmax,preciou,pgan) =
cod
item ::  ItemStock -> Item
item
(cod,item,marca,rubro,prov,umed,cant,cmin,cmax,preciou,pgan) =
item
-- .....
-- y así para cada elemento de la tupla(completar el codigo si
es necesario)
pganancia::ItemStock -> PGan
pganancia
(cod,item,marca,rubro,prov,umed,cant,cmin,cmax,preciou,pgan) =
pgan

-- datos predefinidos (Ejemplos Lab.)
tabla1S:: TStock
tabla1S= [
 (100,"ARROZ GRANO GRANDE", "CONDOR", "Alimentos", 20, "1LT", 8000, 500, 10000, 20, 30),
 (107,"ARROZ GRANO GRANDE", "GALLO", "Alimentos", 20,"1KG", 6000, 200, 8000, 25, 30),
 (200,"ACEITE DE GIRASOL", "NATURA", "Alimentos", 20, "1LT", 9800, 600, 10000, 40,  30),
 (200,"ACEITE DE GIRASOL", "COCINERO", "Alimentos", 20, "1LT", 900, 500, 10000, 30, 30),
 (410,"AGUA MINERAL S/GAS BAJO SODIO", "SER", "Alimentos", 31, "1.5LT", 20, 50, 3000, 10, 35),
 (412,"AGUA SABORIZADA LIMA LIMON", "SER", "Alimentos", 31, "2LT", 1570, 50, 3000, 15, 35),
 (478,"ALFAJOR CHOCOLATE TITA", "TERRABUSI", "Alimentos", 31, "36GR", 900, 200, 5000, 4, 30),
 (479,"ALFAJOR CHOCOLATE RODESIA", "TERRABUSI", "Alimentos", 31, "40GR", 9, 200, 3500, 4, 30),
 (708,"LECHE DESC.PASTEURIZADA", "SERENISIMA", "Alimentos", 31, "1TL", 230, 100, 1200, 20, 30),
 (767,"ARVEJAS SECAS REMOJADAS", "NOEL", "Alimentos", 20, "300GR", 1203, 500, 3000, 10, 30),
 (801,"ANTITRANSPIRANTE ROLL-ON", "ETIQUET", "PERFUMERIA", 20, "60gr", 30, 45, 2000, 25, 30)
 ]
tabla1P :: TProveedor
tabla1P= [
 (20,"Juan Perez","Belgrano 1827, San Luis, 5700, Argentina","2664-786543"),
 (31,"Jose Lopez","Junin 444, Mendoza,5500, Argentina","261-3452677")
 ]
```

Apéndice B

Conceptos teóricos generales sobre Bases de Datos Relacionales (BDR

Una *base de datos relacional* es un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Las bases de datos relacionales se basan en el modelo relacional. En una base de datos relacional, cada fila de la tabla es un registro con un ID único llamado *clave*. Las columnas de la tabla contienen atributos de los datos, y cada registro generalmente tiene un valor para cada atributo, lo que facilita el establecimiento de las relaciones entre los puntos de datos.

El **álgebra relacional** es un conjunto de operaciones que describen paso a paso cómo computar una respuesta sobre las relaciones, tal y como éstas son definidas en el modelo relacional.
Denominada de tipo **procedimental**, a diferencia del Cálculo relacional que es de tipo declarativo.
Describe el aspecto de la manipulación de datos. Estas operaciones se usan como una representación intermedia de una consulta a una base de datos y, debido a sus propiedades algebraicas, sirven para obtener una versión más optimizada y eficiente de dicha consulta.
Cada operador del álgebra acepta una o dos relaciones y retorna una relación como resultado.

En este laboratorio las relaciones están modeladas como listas de tuplas las cuales contienen una cantidad fija de atributos.

La operación unaria **proyección (π)** del Algebra Relacional permite extraer atributos de una relación.
Si $A_1,A_2,..A_n$ son atributos de una relación **R**:

$$\pi_{A1,A3}(R)$$

permite extraer los atributos $\mathbf{A_1}$ y $\mathbf{A_2}$ de la relación **R**.

En la BDRS descripta en este laboratorio, la expresión del ejercicio 4:

```
*LabHask2021> map item tabla1S
```

retorna una lista con el valor del atributo item de todas las tuplas de la relacion tabla1S y modela la operación relacional: $\pi_{Item}(tabla1S)$
La expresión del ejercicio 4**:**

```
*LabHask2021> filter ((==200).codItem)tabla1S
```

modela la operación **selección/restricción (σ)** del Algebra Relacional, que permite seleccionar un subconjunto de tuplas de una relación **R**, todas aquellas que cumplan la(s) condición(es) **p**, esto es:

$$\sigma_p(R)$$

En nuestro caso: $\sigma_{coditem==200}(tabla1S)$

Referencias

https://www.oracle.com/ar/database/what-is-a-relational-database/
https://es.wikipedia.org/wiki/Base_de_datos_relacional
https://es.wikipedia.org/wiki/Algebra_relacional