LABORATORIO 2022 – Programación II

Programación Funcional: Lenguaje Haskell

Completar el código del Apéndice A con las funciones solicitadas.
El archivo con el modulo Haskell debe tener como nombre:

- ***LabH2022PII<Apellido del alumno>*.hs**
  ejemplo: LabH2022PII_Perez.hs

El laboratorio debe ser enviado por mail a la cuenta: **progUNSLCMN@gmail.com**. En tema del mail se deberá colocar el nombre del archivo sin la extensión.
ejemplo: LabH2022PII_Perez

Fecha entrega: hasta el 13/5– 24hs
IMPORTANTE !! El Laboratorio tiene que estar entregado para poder rendir la recuperación del 1er. parcial de Haskell del viernes 14/5

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***).
b) si se justifica, una función recursiva con acumuladores (***mapTupRA).***
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 poder hacerse, justifique.

Ejercicio 2: implementar la función de orden superior **allTup**, de tipo:

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

Esta función toma como argumentos dos predicados y una lista de tuplas de dos elementos y retorna un valor booleno. Si los predicados pasados como primero y segundo argumentos se satisfacen (dan True) para los primeros y segundos elementos de cada tupla respectivamente, en TODAS las tuplas de la lista, la función retorna True, en otro caso retorna False.

Ejemplo de uso:

```
*LabHask2022> allTup (>1) (=='a') [(20, 'a'),(3,'b'),(54,'a')]
*LabHask2022> False
*LabHask2022> allTup (>1) (=='a') [(20,'a'),(3,'a'),(54, 'a')]
*LabHask2022> True.
```

Definir tres variantes:

a) una función recursiva (***allTupR***).
b) si se justifica, una función recursiva con acumuladores (***allTupRA***). En caso de no hacerla explique porqué.
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 poder hacer alguna de las dos funciones solicitadas, justifique.

Ejercicio 3: implementar la función de orden superior ***filterTup***,

```
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***).
b) si se justifica, una función recursiva con acumuladores (***filterTupRA***). En caso de no hacerla explique porqué.
c) dos funciones, usando respectivamente las funciones predefinidas de orden superior foldl (***filteTuprfl***) y foldr (***filterTupfr***) . En caso de no poder hacerse, justifique.

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 **Apendice A**.
- *Ejecutar* en el interprete las siguientes expresiones y en el modulo a entregar, en la zona de comentario indicada en cada caso, describa con palabras que hace cada una.

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

a) Defina la función ***increPU*** que toma como argumento una tabla de Stock, y un nro (que representa un porcentaje) y retorna como resultado la tabla con TODOS los precios unitarios incrementados según el 2do argumento (puede usar las funciones de extracción que figuran en el modelo para extraer el campo precio unitario).

Ejemplo: incrementar los precios unitarios un 10%

```
*LabHask2022> increPU tabla1S 10
[(100,"ARROZ GRANO
GRANDE","CONDOR","Alimentos",20,"1LT",8000,500,10000,22.0,30),
(107,"ARROZ GRANO
GRANDE","GALLO","Alimentos",20,"1KG",6000,200,8000,27.5,30),
(200,"ACEITE DE
GIRASOL","NATURA","Alimentos",20,"1LT",9800,600,10000,44.0,30),
(200,"ACEITE DE
GIRASOL","COCINERO","Alimentos",20,"1LT",900,500,10000,33.0,30),(410,"AGUA
MINERAL S/GAS BAJO SODIO","SER","Alimentos",31,"1.5LT",20,50,3000,11.0,35),
(412,"AGUA SABORIZADA LIMA
LIMON","SER","Alimentos",31,"2LT",1570,50,3000,16.5,35),
(478,"ALFAJOR CHOCOLATE
TITA","TERRABUSI","Alimentos",31,"36GR",900,200,5000,4.4,30),
(479,"ALFAJOR CHOCOLATE
RODESIA","TERRABUSI","Alimentos",31,"40GR",9,200,3500,4.4,30),
(708,"LECHE DESC. PASTEURIZADA",
"SERENISIMA","Alimentos",31,"1TL",230,100,1200,22.0,30),
(767,"ARVEJAS SECAS
REMOJADAS","NOEL","Alimentos",20,"300GR",1203,500,3000,11.0,30),(801,"ANTITRA
NSPIRANTE ROLL ON","ETIQUET","PERFUMERIA",20,"60gr",30,45,2000,27.5,30)]
```

b) Defina la función ***fReponer*** que toma como argumento una tabla de Stock y retorna como resultado una lista con todos los datos de los ítems de stock del almacén cuya existencia en depósito se encuentre por debajo del valor mínimo recomendado para dicho producto.

Ejemplo:

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

Apéndice A

```
-----------------------------------------------
-- Module      : LabH2022
-- Developer   : Apellido y Nombre del alumno
--
-- Programacion II - Laboratorio Haskell 2021
-----------------------------------------------

module LabH2022 where

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

{- ej. 4.a
   Expresion: map item tabla1S
   --- Completar---

   Expresion: filter ((==200).codItem)tabla1S
   --- Completar---

-}

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

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

-------------------------------------------------
-- MODELO SISTEMA DE STOCK DE ALMACEN
-------------------------------------------------

type Cod_Item = Int               -- Codigo Interno del producto
type Item = String                -- Descripcion del producto
type Marca = String               -- Marca
type Rubro = String               -- Rubro
type Cod_Proveedor = Int     -- Codigo Interno del proveedor
type U_Med = String  -- Unidad de Medida:1LT,800GRM, 1500CM3,etc
type Cant_Existente =Int --cantidad de productos en deposito (E)
type V_Min = Int    -- valor en existencia recomendado para
                    -- reposicion (EMin)
type V_Max = Int    -- valor maximo de acopio en deposito (EMax)
type Precio_U = Float     -- precio o valor de compra unitario
type P_Ganancia = Int     -- Porcentaje de ganancia sobre el
                          -- precio de compra
type Nombre = String
type Direccion = String
type Telefono = String
```

```
-- tupla con datos de 1 item de Stock
type Item_Stock = (
   Cod_Item,
   Item,
   Marca,
   Rubro,
   Cod_Proveedor,
   U_Med,
   Cant_Existente,
   V_Min,
   V_Max,
   Precio_U,
   P_Ganancia
  )
-- tupla con datos de 1 proveedor
type Proveedor = (
   Cod_Proveedor,
   Nombre,
   Direccion,
   Telefono
  )

-- Tablas BD
type T_Stock = [Item_Stock]  --Tabla con el Stock de un comercio
type T_Proveedor = [ Proveedor] --Tabla con los proveedores de
un comercio

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

-- datos predefinidos (Ejemplo)
tabla1S:: T_Stock
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 :: T_Proveedor
tabla1P=  [ (20,"Juan Perez","Belgrano 1827, San Luis, 5700,
Argentina","2664-786543"),
            (31,"Jose Lopez","Junin 444, Mendoza,5500,
Argentina","261-3452677")]
```