Paradigmas de Programación – UTN – FRBA
1er cuatrimestre de 2009
Paradigma Funcional – Parcial

Aclaraciones:

- *Esta evaluación es a libro abierto, pueden usar todo lo que tengan en la carpeta y los apuntes que deseen.*
- *Es muy importante poner nombre, nro. de legajo, nro. de hoja y cantidad total de hojas en cada hoja.*
- *Recuerden que la intención es medir cuánto se sabe del paradigma en cuestión; si un ejercicio es resuelto sin utilizar las ideas del paradigma no será considerado. De estas ideas las que más nos interesan son: aplicación parcial, composición y orden superior.*
- *Si no pueden resolver algún ejercicio, suma puntos al menos identificar el tipo de la función que se pedía (cuando no sea dado explícitamente, claro). Además esto ayuda a resolver los ejercicios encadenados.*

Después de varias reuniones que terminaron con algún comensal en malas condiciones, decidimos hacer un sistema para medir cuánto toma cada uno, naturalmente en haskell. Para ello comenzamos pidiendo a los bares que frecuentamos que pusieran a nuestra disposición su carta en un formato adecuado, es decir, una lista de tuplas que contengan, nombre de la bebida, precio y graduación alcoholica. Aquí tenemos un ejemplo que nos envió un bar amigo:

```
carta = [("criadores", 20, 40), ("jb", 25, 40), ("quilmes", 12, 5),
         ("gancia", 12, 5), ("coca", 8, 0)]
```

El siguiente paso será registrar lo que consume cada uno; para ello nuestro sistema representa a cada persona como una tupla que contiene su nombre y la lista de consumos de la noche.

Cada consumo se representa a su vez por una tupla que contiene el nombre de la bebida consumida y el horario. Y el horario es una tupla con dos números enteros: `(hora, minutos)`. Por ejemplo:

```
cesar = ("cesar", [("criadores", (18,55)), ("jb", (19,05))])
guille = ("guille", [("quilmes", (20,50)), ("quilmes",(21,10))])
nico = ("nico", [("coca", (19,0)), ("coca", (20,15))])
```

En el horario de 18 a 21 el bar tiene algunas bebidas con *happy hour*, que se cobrarán a mitad de precio. Las bebidas alcanzadas por la promoción están definidas en una función de la forma:

```
bebidasHappyHour = ["quilmes", "criadores", "gancia"]
```

Ejercicio 1

Desarrollar las siguientes funciones:

a) `precioBase` Dado un nombre de bebida y una carta encontrar su precio.

```
?- precioBase "quilmes" carta
12
```

b) `precioConsumo` Dado un consumo calcular el precio utilizando la función `carta`, teniendo en cuenta el horario en que se realizó dicho consumo.

```
?- precioConsumo ("quilmes", (20,50))
6

?- precioConsumo ("quilmes", (21,10))
12
```

Ejercicio 2

`cuantoPaga` Dada una tupla que representa una persona, devolver cuánto tiene que pagar en función de sus consumos

```
?- cuantoPaga cesar
35 -- 10 + 25 (el criadores está en happy hour).
```

Ejercicio 3

Desarrollar las siguientes funciones:

a) `sanito`, se dice de una persona que es *sanito* si consume únicamente bebidas sin alcohol.

```
?- sanito nico
True
```

b) `sanitos`, recibe una lista de personas con sus consumos y devuelve los nombres de los que se consideran sanitos.

```
?- sanitos [cesar, nico, guille]
["nico"]
```

Ejercicio 4

`terminoArruinado`, recibe una persona y devuelve un booleano indicando si esa persona terminó la noche en mal estado. Se considera que terminó en mal estado una persona que haya tomado dos bebidas blancas (es decir, con más de 35 grados de alcohol) consecutivas pasando menos de 15 minutos entre ambas.

```
?- terminoArruinado cesar
True
```

Ejercicio 5

Desarrollar la función de orden superior `buscarSegun`, que reciba dos funciones. La primera de ellas representará el criterio de búsqueda, debe ser una función de tipo `a -> Bool`. La segunda representará el resultado a devolver y su tupo será `a -> b`.

De esta forma el tipo de `buscarSegun` debe ser `(a -> Bool) -> (a -> b) -> [a] -> b`. Es decir, recibe una lista de `a`, encuentra al primero que cumpla con el criterio especificado y luego lo transforma utilizando la segunda función. Un ejemplo de uso de la función `buscarSegun` puede ser:

```
?- buscarSegun primo doble [20..]
46
```

Esto es, busca el primer primo de la lista y devuelve el resultado de aplicar la función `doble` al número obtenido (naturalmente, el primer primo de la lista es 23).

Luego, utilizar la función buscar según para:

a) Rehacer la función `precioBase` basándose en `buscarSegun`. La nueva implementación debe estar basada únicamente en la función de orden superior, es decir, el ejercicio es encontrar qué hay que poner en lugar de `f` y `g` tal que pueda escribirse:

```
precioBase bebida = buscarSegun f g
```

b) Desarrollar una función que reciba el nombre de una persona y una lista de personas y devuelva cuanto debe pagar.

```
?- cuantoPaga "cesar" [guille, nico, cesar]
35
```