Programación Funcional - Práctico 3

Curso 2017

1. Explique el tipo de las siguientes funciones:
   (a) $min\ x\ y = \textbf{if}\ x < y\ \textbf{then}\ x\ \textbf{else}\ y$
   (b) $paren\ x = \texttt{"("} +\!\!+\ show\ x +\!\!+\ \texttt{")"}$
2. Dada la siguiente definición de tipo:

   $$\textbf{data}\ Semaforo = Verde \mid Amarillo \mid Rojo$$

   ¿Qué falta para que sea posible hacer $(show\ Verde)$?
3. Defina usando recursión explícita la función:
   $merge :: Ord\ a \Rightarrow [a] \to [a] \to [a]$
   que dadas dos listas ordenadas retorna una lista ordenada con el contenido de ambas.
4. Defina las siguientes funciones usando recursión explícita, como *foldr* y como *foldl*:
   (a) $sumSqs :: Num\ a \Rightarrow [a] \to a$, que suma los cuadrados de los elementos de una lista.
   (b) $elem :: Eq\ a \Rightarrow a \to [a] \to Bool$, que determina si un elemento pertenece a una lista.
   (c) $elimDups :: Eq\ a \Rightarrow [a] \to [a]$, que elimina los duplicados adyacentes de una lista. Por ejemplo $elimDups\ [1,2,2,3,4,4,4,3]$ retorna $[1,2,3,4,3]$.
5. Sea $h\ x\ xs = x - sum\ xs$. ¿Cuál de las siguientes afirmaciones es correcta?
   (a) $h\ x\ xs = foldr\ (-)\ x\ xs$
   (b) $h\ x\ xs = foldl\ (-)\ x\ xs$
6. Sea la función $split :: [a] \to ([a],[a])$, que divide la lista en dos listas colocando sus elementos de forma alternada. Por ejemplo $split\ [2,4,6,8,7]$ resulta en $([2,6,7],[4,8])$.
   (a) Implemente *split* como un *foldr*.
   (b) Implemente *split* usando *foldl*.

7. Sea la función $maxInd :: Ord\ a \Rightarrow [a] \to (a, Int)$, que retorna el máximo de una lista no vacía y el índice de su primera ocurrencia. Los índices se comienzan a numerar en 0. Por ejemplo $maxInd\ [8, 10, 6, 10, 10]$ resulta en $(10, 1)$.

   (a) Implemente *maxInd* usando *foldl*.

   (b) Implemente *maxInd* usando *foldr*.

8. Implemente las funciones *takeWhile* y *dropWhile* usando:

   (a) *foldr*.

   (b) *foldl*.

9. Suponga que representamos números naturales como listas de dígitos ordenados de forma descendente según su significación. Por ejemplo [1,2,5] representa al número 125.

   (a) Defina una función $decimal :: [Int] \to Int$, que dado un natural en esta representación compute el entero correspondiente. Escriba dicha función utilizando recursión explícita y como *foldl*.

   (b) Defina una función $repr :: Int \to [Int]$, que dado un natural (de tipo *Int*) retorna su representación. Escriba dicha función utilizando recursión explícita. ¿Se puede hacer utilizando *foldl* o *foldr*?

   (c) Defina una función $sucesor :: [Int] \to [Int]$, que dado un decimal en esta representación compute el siguiente decimal. Escriba dicha función utilizando recursión explícita y usando *foldr*. Se debe trabajar directamente con la representación. No deben usarse las funciones *decimal* ni *repr*.

Nota:

- Cuando decimos "como *foldl*" o "como *foldr*" nos referimos a que la solución debe ser construida solamente en términos del *fold* correspondiente, definiendo los argumentos que dicho *fold* requiera. Ejemplo: *sum* como *foldr*, se define como $sum = foldr\ (+)\ 0$
- Cuando decimos "usando *foldl*" o "usando *foldr*" nos referimos a que la solución contiene un *fold* como parte de ella. Ejemplo: *sumaCero* usando *foldr*, se define como $sumaCero = (0 ==) \circ foldr\ (+)\ 0$.