Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información.
CI–3661 – Laboratorio de Lenguajes de Programación.
Septiembre–Diciembre 2012

Problemario de Semana V: Haskell – Módulos y Monads

1. Implemente el tipo de datos `(Diccionario a)` como un alias para `(Map a String)` e implemente las siguientes funciones:

   a) `consultar :: a -> Diccionario a -> Maybe String`
   Debe devolver la cadena de caracteres asociada a la clave.

   b) `recopilar :: (Ord a) => a -> a -> Diccionario a -> [String]`
   Debe devolver la lista de cadenas de caracteres asociadas a las claves comprendidas entre las dos claves propuestas.

   c) `sinonimos :: String -> Diccionario a -> [a]`
   Debe devolver la lista de claves que asocian con la misma cadena de caracteres.

2. Evalúe las siguientes expresiones en Haskell:

   - `Just "Haskell" >> Just "Rocks!"`
   - `Nothing >> Just "Rocks!"`
   - `Just "Haskell" >> Nothing`
   - `Just 21 >>= return`
   - `Just 21 >>= return . (*2)`
   - `Nothing >>= return . (*2)`
   - `[] >> [1..5]`
   - `[1] >> [1..5]`
   - `[1..5] >> [1..5]`
   - `[1..5] >>= return . (*2)`
   - `getLine >>= putStrLn`
   - `getLine >>= putStrLn . reverse`
   - `sequence_ $ map putStrLn ["Haskell", "Rocks!", "ci3661"]`
   - `sequence ["pP", "iI", "oO", "!?"]`
   - `map ("Il polcino " ++) $ sequence ["pP", "iI", "oO", "!?"]`
   - `foldr (const sequence) [[1..3]] [1]`
   - `foldr (const sequence) [[1..3]] [1,1]`
   - `foldr (const sequence) [[1..3]] [1,1,1]`
   - `take 5 $ map (foldr (const sequence) [[1..3]]) (zipWith replicate [1..] [1..])`

3. Proponga una instancia completa de la clase `Monad`, para el tipo de datos `Either a b`, que cumpla con las leyes monádicas. (Extra: Una vez definida, proponga una instancia de la clase `MonadPlus` para el mismo tipo, que cumpla con las leyes de `MonadPlus`).