Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información.
Laboratorio de Lenguajes de Programación I (CI-3661)
Septiembre - Diciembre, 2010

Proyecto I de Programación Funcional
Codificación y Decodificación de Mensajes

En este proyecto, ud. implementará un módulo en Haskell que trabaje con codificación y decodificación de mensajes. La idea principal del proyecto es elaborar un programa que obtenga, de dos listas, la posible función de codificación que permita traducir los mensajes del origen al destino.

Para poder realizar esto, definimos el siguiente tipo de datos que representa los mensajes:

```
type Mensaje a = [a]
```

y la siguiente, que representaría la función de codificación:

```
type Codificador a b = [(a, b)]
```

Funciones a implementar

Para este proyecto, ud. va a implementar las siguientes funciones:

- La función:

  ```
  obtenerCodificador :: Mensaje a -> Mensaje b -> Codificador a b
  ```

  donde `obtenerCodificador origen destino` es la codificación que se puede obtener de `origen` a `destino`. Por ejemplo:

  ```
  obtenerCodificador "hola" "aloh" = [('h', 'a'),('o', 'l'),('l', 'o'),('a', 'h')]
  obtenerCodificador "abcd" [1, 2, 3, 4] = [('a', 1),('b', 2),('c', 3),('d', 4)]
  ```

  El orden en que se devuelve el codificador no importa.

- La función:

  ```
  codificadorValido :: Codificador a b -> Bool
  ```

  donde `codificadorValido c` es cierto si `c` es un codificador válido. Un codificador es válido si para un par (x, y) que exista en el compilador no exista otro par que contenga a x o, en el caso que sea repetido, el segundo elemento del par es y.

- La función:

  ```
  inversoCodificador :: Codificador a b -> Codificador b a
  ```

  donde `inversoCodificador c` es el codificador inverso de c, es decir, para todo (x, y) en `c`, está el par (y, x) en `inversoCodificador c`.

- La función:

  ```
  codificadorInvertible :: Codificador a b -> Bool
  ```

  donde `codificadorValido c` es cierto si `c` es un codificador invertible. Un codificador invertible es aquel que sea válido y además el codificador inversible es válido.

- La función:

  ```
  codificarMensaje :: Mensaje a -> Codificador a b -> Mensaje b
  ```

  donde `codificarMensaje cod msg` es el mensaje resultante de aplicar `cod` a `msg`. Por ejemplo:

  ```
  codificarMensaje [('h', 'a'),('o', 'l'),('l', 'o'),('a', 'h')] "hallo" = "ahool"
  ```

Este proyecto debe ser implementado usando solamente funciones del preludio de Haskell. No se permite usar librería alguna para su solución del proyecto.

Datos administrativos

- El proyecto debe ser entregado, a su correspondiente encargado de sección, a más tardar el viernes de semana 3, a mas tardar a las 11.59 pm.
- Tiene una ponderación de 12 pts.

C. A. Pérez, W. Pereira