Facultad de Ciencias - UNAM
Lógica Computacional 2020-2
Práctica 1: Introducción a Haskell

Favio E. Miranda Perea
Alejandra Krystel Coloapa Díaz
Pedro Juan Salvador Sánchez Pérez

31 de enero de 2020
fecha de entrega: 10/02/2020

En el archivo Binarios.hs se encuentra definido un tipo de datos:

```
Binario = U | Cero Binario | Uno Binario
```

Este tipo de datos representa los números binarios mayores o iguales a uno.
Observa que esta representación tiene al bit menos significativo como constructor, es decir:

U: Representa al dígito 1.

Cero x: Representa al binario x0 donde x es un binario.

Uno x: Representa al binario x1 donde x es un binario.

Nota que la representación del tipo de dato esta al revés de la forma normal de escribir un número binario.

1. Ejercicios

1. (1 punto) Crea una instancia de la clase `Show` para que muestre los binarios en su notación usual, utiliza la función show.
   Ejemplos
   - `Practica1*>Cero (Uno U)`
     `110`
   - `Practica1*>Uno (Cero U)`
     `101`
2. (2 puntos) Define una función llamada sucesor la cual reciba un binario y devuelva su sucesor.
   Ejemplos

- `Practica1*>Sucesor (Uno U)`
  `100`
- `Practica1*>Sucesor (Cero U)`
  `11`

3. (2 puntos) Define una función llamada suma la cual reciba dos binarios y devuelva la suma de estos.
   Ejemplos

   - `Practica1*>suma (Uno U) (Cero U)`
     `101`
   - `Practica1*>suma (Uno U) (Uno U)`
     `110`

4. (2 puntos) Define una función llamada producto la cual reciba dos binarios y devuelva el producto de estos.

   - `Practica1*>producto U U`
     `1`
   - `Practica1*>producto (Cero U) (Cero U)`
     `100`
   - `Practica1*>producto (Uno U) (Uno U)`
     `1001`

5. (1 punto) Define una función llamada natBinLista la cual reciba un número entero (mayor o igual a 1) y devuelva su notación binaria en forma de lista.

   - Practica1*¿natBinLista 10
     `[1,0,1,0]`
   - `Practica1*>natBinLista 0`
     `[]`
   - `Practica1*>natBinLista 101`
     `[1,1,0,0,1,0,1]`

6. (2 puntos) Define una función llamada sumaBinLista la cual reciba dos listas de enteros representado números en notación binaria y devuelva su suma de tipo Binario.
   Sugerencia: Crea una función auxiliar que convierte un número binario en forma de lista a un tipo Binario y después utiliza la función suma definida previamente.

   - `Practica1*>sumaBinLista [1,0,1] [1,0]`
     `111`
   - `Practica1*>sumaBinLista [1,0,1] [1,1,1,1]`
     `10100`

2. Puntos extra

1. (1 punto) Define una función llamada natABin la cual reciba un número entero (mayor o igual a uno) y devuelva su notación en binario.

- `Practica1*>natABin 25`
  `11001`
- `Practica1*>natABin 0`
  `** Exception: Solo numeros mayores a 0.`

2. (1 punto) Define una función llamada *binANat* la cual reciba un número binario y devuelva su representación en número entero.

   - `Practica1*>binANat (Uno (Uno (Cero U)))`
     `11`

3. (2 puntos) Define una función llamada predecesor la cual reciba un binario y devuelva el binario anterior.

   - `Practica1*>predecesor U`
     `1`
   - `Practica1*>predecesor (Cero U)`
     `1`
   - `Practica1*>predecesor (Uno U)`
     `10`

3. Para pensar

Observa que se utilizaron dos formas de representar números binarios: definiendo un tipo y utilizando los tipos primitivos brindados por el lenguaje. ¿Cuál consideras mejor implementación? Esta es una pregunta importante al momento de definir un lenguaje de programación.

4. Sugerencias generales

- Definir funciones y hacer ejemplos en papel (coloquialmente dicho: programar en papel).
- Documentación de *Haskell* : https://wiki.haskell.org/.
- El mejor tutorial de Haskell : http://aprendehaskell.es/.
- Se pueden utilizar ideas que se encuentren en internet, pero eso: **ideas**.