Ejercicio Semanal 1

Lógica Computacional, 2018-2
Facultad de Ciencias, UNAM

Noé Salomón Hernández Sánchez
no.hernan@gmail.com

María del Carmen Sánchez Almanza
carmensanchez@ciencias.unam.mx

Albert Manuel Orozco Camacho
alorozco53@ciencias.unam.mx

1 de febrero de 2018

1. Objetivo

Que el alumno empiece a conocer y/o ejercitar sus habilidades en programación funcional. Se familiarizará, además, con la sintaxis y el modo de ejecución del lenguaje de programación `Haskell`. Se trabajará con listas, recursión, tipos de datos `Enum` (recursivos) y flujos de control.

2. Ejercicios

El esqueleto de código del ejercicio semanal se encuentra en `https://github.com/alorozco53/LabLogComp-2018-2/tree/ejsem1`.

2.1. Listas

1. Proponga una función `myReverse` que calcule la reversa de una lista. No utilice la función `reverse` del *Preludio*.

2. Escriba una función `myTake` que, dado un entero k y una lista l, devuelva el prefijo de tamaño k de l. Por ejemplo,

```
*EjercicioSemanal1> myTake 3 [5,6,7,10,12,34,0]
[5,6,7]
```

No utilice la función `take` del *Preludio*.

3. Elabore una función `myCount` que, dado un elemento de cierto tipo y una lista del mismo tipo, cuente cuántas veces se repite dicho elemento en la lista.

4. Escriba una función `myFreq` que, dada una lista l, devuelva la lista de 2-tuplas, donde, para cada tupla, la primera entrada es un elemento de l y la segunda es el número de veces que dicho elemento se repite en l. Se otorgarán $\frac{1}{4}$ de puntos extras por una implementación *elegante*. Por ejemplo:

```
*EjercicioSemanal1> myFreq "alonzo church"
[('a',1),('l',1),('o',2),('n',1),('z',1),('o',2),
 (' ',1),('c',2),('h',2),('u',1),('r',1),('c',2),('h',2)]
```

5. Dada una lista l y dos enteros i, j, devuelve la sublista que empieza en el i-ésimo elemento de l y termina en el $j-1$-ésimo. Esto es, análogo del `list[i:j]` de Python.

6. Dada una lista de cierto y un elemento del mismo tipo, devuelve la lista de listas formada por todas las sublistas que preceden y suceden las apariciones del elemento Por ejemplo:

```
*EjercicioSemanal1> split "the quick brown fox" ' '
["the","quick","brown","fox"]
```

   Análogo al list.split() de Python.

7. Considere el siguiente algoritmo de compresión de cadenas: *dada una palabra de tamaño k alfanumérica (sin espacios entre carácter), mantenga únicamente los primeros $\lfloor \frac{k}{2} \rfloor$ elementos de ésta; realice esto para cada palabra (en orden) de la cadena en cuestión; finalmente, concatene todos los prefijos restantes.* Proponga una función `dumbCompress` que implemente el algoritmo anterior. Por ejemplo:

```
*EjercicioSemanal1> dumbCompress "la science n'a pas de patrie"
"lscinpdpat"
```

2.2. Números naturales

Considere el tipo de datos de números naturales

```
data Nat = Zero | Succ Nat
```

para la elaboración de lo siguiente.

8. Escriba una función `sumNat` que sume dos números naturales.

9. Proponga una función `prodNat` que multiplique dos números naturales.

10. Elabore una función `powerNat` que eleve un número natural a la potencia de otro.

11. Escriba una función `eqNat` que decida si dos números naturales son iguales. No utilice la función `(==)` auto-implementada en la declaración del tipo `Nat`.

12. Dé una función `greaterThan` que, dados números naturales n y m, decida si n es mayor (estricto) que m.

13. Escriba una función que convierta un número natural de tipo `Nat` en un entero de tipo `Int`.

14. Escriba una función que convierta un entero de tipo `Int` en un número natural de tipo `Nat`.

15. Elabore una función `throwRev` que, dado un `Nat` n y una lista l, quite los n últimos elementos de l.

3. Entrega

La fecha de entrega es el próximo **jueves 8 de febrero de 2018** por la plataforma de *Google Classroom* del curso y siguiendo los lineamientos del laboratorio.