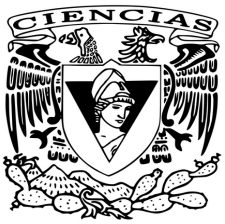

Lógica Computacional 2017-2
Práctica 1: Introducción a `Haskell`

Lourdes del Carmen González Huesca
Roberto Monroy Argumedo
Fernando A. Galicia Mendoza

Facultad de ciencias, UNAM

Fecha de entrega: Martes, 14 de febrero del 2017
¡Feliz día del amor y la amistad! ♡

Esta práctica deberá ser entregada individualmente.

1. Implementación

1.1. Fórmula general de segundo grado

Una de las primeras fórmulas que se enseñan en bachillerato es la ecuación general de segundo grado, cuyo fin es encontrar las raíces de polinomios de segundo grado en un campo K.

La primera parte de esta práctica es crear un programa que reciba un polinomio (en $\mathbb{R}$) de segundo grado y devuelva sus posibles soluciones. Para lograr esto crea un archivo llamado `Chicharronera`, considera los siguientes tipos de datos:

```
-- | C. Tipo de datos que representa los numeros complejos.
type C = (Double,Double)

-- | P2. Tipo que representa los polinomios de segundo grado
-- en el campo de los numeros reales, es decir,
-- (a,b,c) representa ax^2+bx+c.
type P2 = (Double,Double,Double)

-- | R. Tipo que representa las dos posibles raices de un
-- polinomio de segundo grado.
type R = (C,C)
```

Y realiza los siguientes ejercicios:

1. (0.5 puntos) Define una función que reciba un polinomio de segundo grado y determine si este tiene raíces reales, es decir, `f p = True` syss el polinomio `p` sus dos raíces son números reales.

2. (0.5 puntos) Define una función que reciba un polinomio de segundo grado y devuelva el par consistente en las dos posibles raíces de un polinomio de segundo grado, es decir, `f p = (r1,r2)` donde `r1` y `r2` son las raíces del polinomio `p`.
   Sugerencia: Definir dos funciones auxiliares que obtengan la raíz positiva y negativa, respectivamente.

1.2. El cifrado César

En *Vida de los césares*, Suetonio escribió:

Si tenía que decir algo confidencial, lo escribía usando el cifrado, esto es, cambiando el orden de las letras del alfabeto, para que ni una palabra pudiera entenderse. Si alguien quiere decodificarlo, y entender su significado, debe sustituir la cuarta letra del alfabeto, es decir, la D por la A, y así con las demás.

Lo que el autor describe es el sistema de cifrado creado por Julio César, este cifrado fue creado con el único fin de que sus enemigos no pudiesen saber sus estrategias militares.

En esta parte de la práctica tu trabajo es implementar el cifrado César, para esto crea un archivo llamado `Cesar` y realiza los siguientes ejercicios:

1. (0.5 puntos) Define una función que reciba una letra mayúscula del alfabeto latino (es decir, un elemento de tipo `Char`) y devuelva el entero correspondiente a la posición de la letra en el alfabeto latino, es decir, A = 0, B = 1, ..., etc.
   A esta función nómbrala `alf`.
   Observación: En caso de no recibir una letra mayúscula devolver error.

2. (1 punto) Define una función que reciba un entero que represente la posición de una letra de acuerdo al alfabeto latino y devuelva la posición de acuerdo a la función hash descrita por Suetonio.

3. (1 punto) Define una función que reciba una palabra y devuelva su cifrado César.
   Observación: Si la palabra tiene minúsculas, convertirlas a mayúsculas.

1.3. Números binarios

Dado un número natural, existe una conversión a número binario y viceversa. Usualmente los números binarios son representados en `Haskell` por la siguiente expresión:

```
data Binario = [Int]
```

Donde es necesaria la suposición que la lista sólo contenga 0's y 1's. Esto resulta incoveniente a la hora de ejecución del programa, ya que, se tendría que hacer un sistema de seguridad que no permita el ingreso a la lista de otro tipo de números enteros.

Para resolver esto es posible definir un tipo de datos capaz de representar los números binarios, para obtener este tipo de datos, considera la siguiente gramática:

$$BinarioPos \ ::= \ U \mid Cero \ BinarioPos \mid Uno \ BinarioPos$$

Donde:

- U: Representa el dígito uno.
- $Cero\ x$: Representa el binario $x0$ donde x es un binario.
- $Uno\ x$: Representa el binario $x1$ donde x es un binario.

En esta parte de la práctica debes brindar un tipo de datos que represente los números binarios y definir ciertos operadores sobre este tipo de datos, para esto realiza los siguientes ejercicios:

1. (0.5 puntos) Define un tipo de datos que represente a $BinarioPos$, bríndale el mismo nombre y agrega la expresión `deriving Show`.
2. (1 punto) Define una función recursiva que dado un número binario, devuelva su sucesor.
3. (1 punto) Define una función recursiva que dados dos números binarios, devuelva su suma.
4. (1 punto) Define una función recursiva que dados dos números binarios, devuelva su producto.

2. Teoría

La siguiente sección deberá ser entrega impresa[1] al inicio de la sesión de ayudantía:

1. (1 punto) Demuestra la correctud de la función `mapeo` definida en `Haskell`, es decir:
   Para cualquiesquiera tipos `a` y `b`, función `f` que dado un elemento de tipo `a` devuelve un elemento de tipo `b` y lista `l` de tipo `a`, se cumple lo siguiente:
   Si `l = [x1, x2,..., xn]` con `xi` elemento de tipo `a`, entonces

   ```
   mapeo f l = [f x1, f x2, ..., f xn]
   ```

   Sugerencia: Inducción sobre la estructura de listas.
2. (1 punto) Demuestra que la función `alf` es parcial y no inyectiva.
3. (1 punto) Demuestra que los operadores de suma y sucesor de números binarios, definidos en tu práctica, cumplen la siguiente propiedad:
   Dado `x` elemento de tipo `BinPos` se cumple que:

   ```
   sucesor x = suma U x = suma x U
   ```

   Sugerencia: Análisis de casos sobre `x`.

[1] Puede ser en LaTeXo a mano.

3. Reglas

- Únicamente se puede importar la biblioteca `Char`.
- No se puede utilizar la función `fromEnum`.
- Para los números binarios, no se puede hacer la conversión a números enteros y viceversa.

El programador funcional suena como un monje medieval, negándose los placeres de la vida con la esperanza de hacerse virtuoso. - Hughes, John