Programación Declarativa 2020-2
Facultad de Ciencias UNAM
Tarea 2: The Imperative is Dark and Full of Terrors

Favio E. Miranda Perea Javier Enríquez Mendoza

Fecha de entrega: 21 febrero de 2020

Parte Teórica

1. Demuestra las siguientes propiedades

```
sum . map double = double . sum
sum . map sum = double . concat
sum . sort = sum
```

   en donde `double` se define de la siguiente manera:

```
double :: Integer -> Integer
double x = 2 * x
```

   y `sum`, `map`, `sort` y `concat` son las definidas en el `Prelude` de Haskell.

2. En Haskell la función `take n` toma los primeros `n` elementos de una lista, mientras que `drop n` regresa la lista sin los primeros `n` elementos de ésta. Demuestra o da un contraejemplo de cada una de las siguientes propiedades.

```
take n xs ++ drop n xs = xs
take m . take n = take (min m n)
map f . take n = take n . map f
filter p . concat = concat . map (filter p)
```

3. Consideremos la siguiente afirmación

```
map (f . g) xs = map f $ map g xs
```

   (a) ¿Se cumple para cualquier `xs`? Si es cierta bosqueja la demostración, en caso contrario ¿que condiciones se deben pedir sobre `xs` para que sea cierta?

   (b) Intuitivamente ¿qué lado de la igualdad resulta mas eficiente? ¿Esto es cierto incluso en lenguajes con evaluación perezosa? justifica ambas respuestas.

Parte Práctica

1. En Teoría de Conjuntos los números naturales se definen como el conjunto de todos los naturales menores a él. Es decir

$$S(n) = \{0, 1, 2, ..., n\}$$

Con esta idea en mente define la función `gtrPower2 n` que encuentra la mayor potencia de 2 en la representación con conjuntos de `n`. Otra forma de verlo es encontrar la potencia de 2 mas grande menor a `n`. La firma de la función es la siguiente:

```
gtrPower2 :: Int -> Int
```

Ejemplos

```
gtrPower2 3 = 2
gtrPower2 8 = 4
```

2. Definir la función `inarow` que recibe una lista de elementos comparables y regresa el mayor número de apariciones consecutivas del mismo elemento. La firma de la función es:

```
inarow :: (Eq a) => [a] -> Int
```

Ejemplos

```
inarow "aabaaabbab" = 3
inarow [1,2,3,3,4,5] = 2
```

3. Cuando el matemático Ingles G.H. Hardy visito al gran matemático Indio Srinivasan Ramanujan, que estaba internado en un hospital en Londres. Durante la conversación Hardy dijo que el taxi en el que había llegado tenía el número 1729, un número que le parecía muy aburrido. "Not at all" dijo Ramanujan en un ingles casi incomprensible, y explico que 1729 era el primer número que podía expresarse como la suma de dos cubos en diferentes formas $1^3 + 12^3 = 1729 = 9^3 + 10^3$.

Define una función dado un parámetro entero `n` regrese una lista con tuplas de la forma `(a,b,c,d)` tal que $0 < a, b, c, d \leq n$ y $a^3 + b^3 = c^3 + d^3$.

Hay que tener especial cuidado pues la ecuación $a^3 + b^3 = c^3 + d^3$ puede escribirse de 8 formas distintas y debería aparecer una única vez en nuestra lista.

La firma de la función es:

```
ramanujan :: Int -> [(Int,Int,Int,Int)]
```

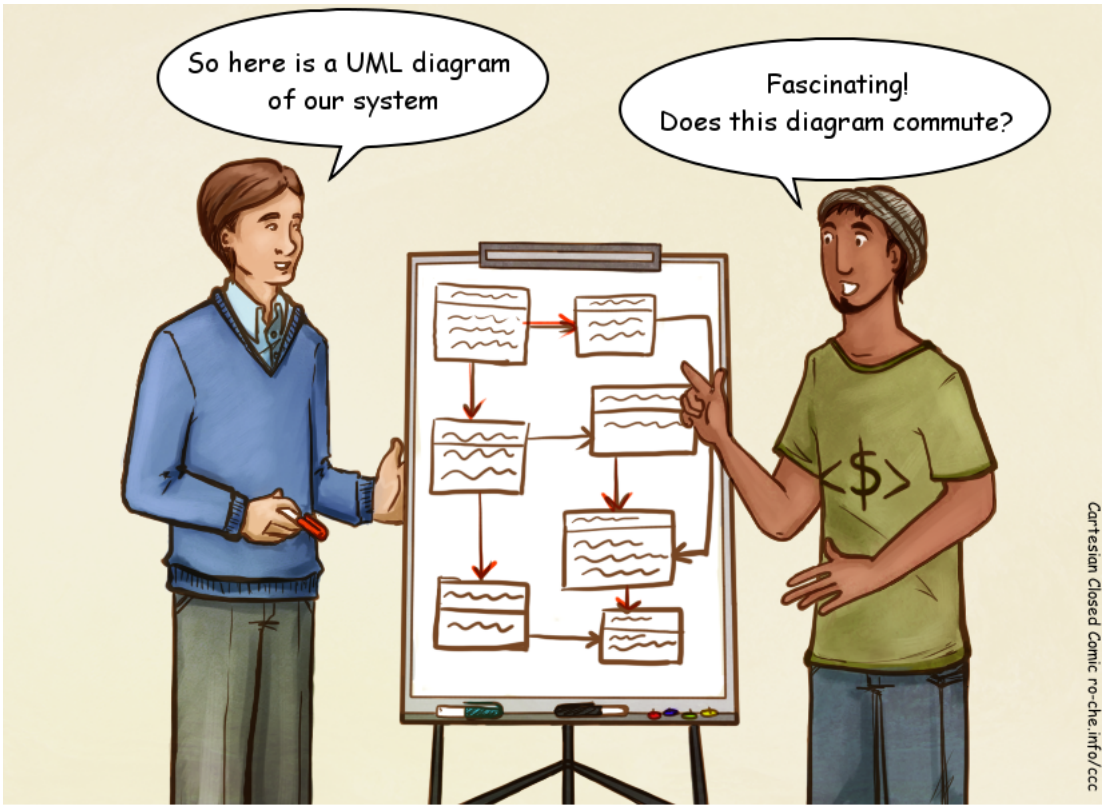
Ejemplos

```
ramanujan 4 = [(3,1,2,2), (4,0,4,0)]
```

El orden en el que parezcan las tuplas no importa mientras todas estén en la lista.

Entrega

- La tarea puede entregarse de forma individual o en parejas.
- La parte práctica de la tarea se entrega a través de Slack:
  - Todos los archivos deben tener al principio como comentario los nombres de los alumnos.
  - Si la tarea se realizo en parejas crear un canal privado con el ayudante y los miembros del equipo para la entrega.
  - Si se realizo de forma individual enviarla en un mensaje directo al ayudante.
  - **No** deben comprimir los archivos, en caso de tener mas de uno enviarlos por separado.
- La parte teórica se puede entregar en PDF (obtenido a partir de LaTeX) junto con la parte práctica o a mano en la **ayudantia** previa a la fecha de entrega.



Transition from Haskell to Java can be awkward