Universidad Nacional Autónoma de México
Facultad de Ciencias
Programación Declarativa

Tarea 2 Parte Teórica

Ángel Iván Gladín García
No. cuenta: 313112470
angelgladin@ciencias.unam.mx

24 de Febrero 2019

1. Demuestra las siguientes propiedades

```
sum . map double = double . sum
sum . map sum = sum . 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 ¿qué condiciones se deben pedir sobre `xs` para que sea cierta?
   TODO

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