Programación declarativa. Tarea 2
The Imperative is Dark and Full of Terrors

Juan Alfonso Garduño Solís
Emiliano Galeana Araujo

Facultad de ciencias, UNAM

Fecha de entrega: Lunes 24 de febrero de 2020

1 Demostraciones de propiedades

-
```haskell
sum . map double = double . sum
```

-
```haskell
sum . map sum = sum . concat
```

-
```haskell
sum . sort = sum
```

Donde, `double` se define de la siguiente manera:

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

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

2 Función take

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 esta. Demuesrta o da un contraejemplo:

```
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 Función map

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?

(b) Intuitivamente, ¿Qué lado de la igualdad resulta más eficiente? ¿Esto es cierto incluso en lenguajes con evaluación perezosa? Justifica tu respuesta.