Programación Funcional

Ejercicios de Práctica Nro.1

Modelo de cómputo funcional

Aclaraciones:

- Los ejercicios siguen un orden de complejidad creciente, y cada uno puede servir a los siguientes. No se recomienda saltear ejercicios sin consultar antes a un docente.
- Recordar que se pueden aprovechar en todo momento las funciones ya definidas, tanto las de esta misma práctica como las de prácticas anteriores.
- Probar todas las implementaciones, al menos en una consola interactiva.
- Es sumamente aconsejable resolver los ejercicios utilizando primordialmente los conceptos y metodologías vistos en clase, dado que los exámenes de la materia evalúan principalmente este aspecto. Para utilizando formas alternativas al resolver los ejercicios consultar a los docentes.

Ejercicio 1) Escribir ocho expresiones que denoten al número `4`, diferentes a las ya vistas (`4`, `2+2`, `3+1`, `doble 2`, `doble (1+1)`). Al menos seis deben usar funciones (o sea, no ser de la forma de una cuenta de sumas o restas), de las cuales: al menos dos deben usar una expresión lambda, tres deben usar `doble` y una debe usar `cuadruple`.

Recordar que las definiciones dadas para `doble` y `cuadruple` son las siguientes

```
doble x = x + x
cuadruple x = 4*x
```

Ejercicio 2) Mostrar que la expresión `doble (doble 2)` puede reducirse de otras formas que la vista en clase. **Sugerencia**: considerar el `doble` más externo, en lugar del interno.

Ejercicio 3) Reducir `cuadruple 2`, y `cuadruple (cuadruple 2)`. ¿Alguna de ellas puede hacerse de más de una forma?

Ejercicio 4) Definir las funciones `triple`, `succ`, `sumarDos`.

Ejercicio 5) Comprobar que `twice succ = sumarDos`.

Ejercicio 6) Dar tres ejemplos de pares de expresiones que sean equivalentes entre sí, donde al menos una de ellas sea una expresión atómica, pero no estén vinculadas por reducción (además de `cuadruple` y `\x -> 4*x`). Dos de ellas deben no contenter lambdas y al menos una debe contener una expresión no atómica.

Ejercicio 7) Realizar la reducción completa de `((twice twice) doble) 3`.

Ejercicio 8) Dar expresiones lambda que sean equivalentes a las siguientes expresiones:

a. `triple`

b. **`succ`**
c. **`sumarDos`**
d. **`twice`**
e. **`twice twice`**

Ejercicio 9) Estudiar cómo es la sintaxis y cómo se utilizan expresiones basadas en **`let`**, **`if-then-else`**, **`case`** y **`where`**. Reescribir las siguientes funciones sin el uso de **`let`**, **`where`** o **`if-then-else`** cuando sea posible.

a. **`f x = let (y,z) = (x,x) in y`**
b. **`f (x,y) = let z = x + y in g (z,y) where g (a,b) = a - b`**
c. **`f p = case p of (x,y) -> x`**
d. **`f = \p -> let (x,y) = p in y`**