

Paradigmas de programación UNSAM 2020

Orden Superior

- Funciones de orden superior tienen un propósito de delegar la responsabilidad a otras funciones que utilizan, para el propósito de la estructura dada. Un ejemplo es la familia de orden superior Fold, que respetan una estructura y tienen un objetivo: Combinar los elementos usando una función para transformar la estructura y reducirla o transformarla. En el ejemplo se ve en el punto 1.4 como dado una función `ventasCriterio criterio`, y dada la serie de criterios podemos retornar la transformación pedida

Evaluación diferida o perezosa (lazy)

- La idea de evaluación perezosa es ir evaluando los elementos a medida que los voy necesitando.

Esto se puede ver en la superVenta, donde estoy usando el concepto de listas infinitas para crear una variable según un parámetro.

Utilizando una lista Potencialmente infinita creo una finita, con los requerimientos dados.

Composición

La idea de la composicion es mantener un orden de la manera que en terminos matematicos es mantener la imagen de la funcion en el dominio de la otra

- $F \circ G / g(f(a)) = (g \circ f) x$

Utilice la composicion en casi la mayoria de las funciones

En haskell la implementacion es $g (f x) = (g.f) x$

Ejemplo Coloquio: `ventasMes (m,anio) = (((==m).mes).fecha)`

Particularmente Tambien utiliza point Free

Listas por comprensión, lambdas

- Haskell esta basado en un sistema de reglas de trasformacion de expresiones
- La idea es la de funcion anonima que se puede Codificar:
 $\lambda x \rightarrow x$ Es la función Identidad
En el caso del Coloquio Se puede ver en varias ocasiones De la forma ($\lambda \text{sum variable} \rightarrow \text{sum} + \text{variable}$)
Junto a la variable foldl, son muy utiles para la familia fold map o filter, cuando no tenemos necesidad de reutilizar la función
- Listas por compresion: es una variable de ver las funciones como map o filter, con la comodidad que podemos ver de donde sale la lista y las variables derecha a izquierda Hasta la funcion aplicada
[listaComponentes venta | venta \leftarrow ventas]

Recursion, orden superior

- En particular es para notar la función `MaximoSegun` funcion lista
Recursion : caso base + recursion otroCaso

*se puede tener mas de un caso base

Esta funciona de manera recursiva con un caso base, lo interesante de esta función es que puedo calcular recursivamente el máximo según una aplicación que me sea conveniente, lo unico a tener en cuenta es que el retorno sea tipo `Eq`, para poder comprarla.

En este caso la funcion `montoVentas` que reutiliza la funcion `precio de maquina del punto 1`