Paradigmas de Lenguajes de Programación

Tomás Agustín Hernández

Recordando Haskell

Para ejecutar un archivo hay que instalar GHCI. Una vez instalado, nos paramos en la terminal en el directorio donde está el archivo que queremos ejecutar.

- Cargar archivo: :l nombreArchivo
- Ver tipo: :type tipo
- Ejecutar funcion: funcion parametro1 parametro2...
- Recargar archivo: :r
- Si necesitamos hacer cálculos para mandar un parámetro, usar paréntesis: Ej.: otherwise = n * factorial(n-1)
-

Maybe

El Maybe se utiliza en Haskell para recibir/devolver respuestas condicionales que pueden ser de un tipo u otro.

Se define como $data\ Maybe\ a\ =\ Nothing\ |\ Just\ a$

Ej.: $devolverFalsoSiVerdadero\ :\ Bool \rightarrow Prelude.Maybe\ Bool$

El Maybe deja la puerta abierta a un valor posible "Nothing". Entonces tenemos dos casos: Si me envian un True devuelvo False (tipo bool), caso contrario, devuelvo Nothing.

Either

El Either se utiliza en Haskell para poder recibir/devolver un parámetro que podría ser de un tipo u otro.
Se define como $data\ Either\ a\ b\ =\ Left\ a\ |\ Right\ b$

Para poder saber qué operación hacer según el tipo literalmente en código usamos (Left valor) o (Right valor).

Ej.: $devolverRepresentacionIntBool\ ::\ Either\ Int\ Bool\ \rightarrow\ Int$

Si es un entero, devuelvo ese mismo entero porque no hago nada. Eso lo hacemos con $Left(a)\ =\ a$, ahora, si el tipo es booleano tengo que decir explícitamente la respuesta según su valor. Es decir, $Right(False)\ =\ 0$ sino, $Right(True)\ =\ 1$.

Declaración de tipos en Haskell

Se utiliza $data\ nombretipo\ tipo\ =\ Tipo\ 1\ |\ Tipo\ 2$ El | se interpreta como .º bien"

Árboles Binarios

Es un tipo (para mi parecer) meramente recursivo.
$data\ AB\ a\ =\ Nil\ |\ Bin\ (AB\ a)\ a\ (AB\ a)$ Nótese que es algo re contra recursivo, porque para definir el tipo de AB a decimos que es un Bin que a su vez es de AB a y a su vez AB a es otro árbol binario. Veamos unos ejemplos de esto

- Bin (Nil) Nil (Nil): es el árbol que no tiene ni siquiera raíz. Y nótese que en cada paréntesis es importante indicar el Nil pues es la forma de que el tipado de Haskell nos lo acepte.
- Bin (Bin Nil 3 Nil) 4 (Bin Nil 6 Nil): Es el árbol que comienza con un Nodo raíz que tiene el valor de 4. El hijo izquierdo del Nodo con valor 4 es otro árbol binario que tiene como valor 3 en su nodo y no tiene hijos. El hijo derecho del Nodo con valor 4 es otro árbol binario que tiene como valor 6 en su Nodo y no tiene hijos.

Y así sucesivamente, veamos un dibujo para tener algo más visual.
El siguiente árbol binario: Bin (Bin (Bin Nil 2 Nil) 3 Nil) 4 (Bin (Bin Nil 5 Nil) 6 Nil) representa el siguiente: