Programación en Haskell

Programación Declarativa

Maximiliano Eschoyez

2021

El lenguaje Haskell

- ¿Por qué Haskell? $\rightarrow$ Presentación
- Historia sobre programación funcional
- Sitio oficial $\rightarrow$ www.haskell.org
- Bibliografía
  1. Documentación oficial
  2. Glasgow Haskell Compiler (GHC)
  3. Prelude Tour
  4. Learn You a Haskell for Great Good!
  5. RAZONANDO CON HASKELL. Un curso sobre programación funcional
- Breve intro para probar

Herramientas para trabajar

GHC Glasgow Haskell Compiler

- Haskell Platform

vscode Visual Studio Code con las extensiones

- *Haskell*
- *Haskell Syntax Highlighting*

Lenguaje Haskell

Se puede compilar

- Archivos estándar tiene extensión `.hs`
- Archivos *literate* tiene extensión `.lhs`

Se puede usar en forma interactiva

GHCi - Entorno Interactivo

Breve introducción para usar GHCi

- `:h` o `:help` lo obvio
- Configuraciones útiles
  - `:set +t` muestra tipos de datos
  - `:set +s` muestra estadísticas de ejecución
- `:l <arch>` o `:load <arch>` carga el archivo y lo interpreta
- `:q`, `:quit` o `Ctl+D` para salir
- No se pueden usar instrucciones multilínea directamente. Se pueden escribir
  - separadas por punto y coma

```
signo :: (Integral a) => a -> a; signo x = mod x
```

  - encerradas entre llaves

```
:{
signo :: (Integral a) => a -> a
signo x = mod x 2
:}
```

Tipos de Datos

- `Num` ⇒ Es un valor numérico
- `Real` ⇒ Es un valor numérico real
- `Fractional` ⇒ Es un valor numérico fraccional
- `Integral` ⇒ Es un valor numérico entero
  - `Int` ⇒ Limitado
  - `Integer` ⇒ Virtualmente infinito
- `Floating` ⇒ Es un valor de punto flotante
  - `Float` ⇒ Precisión simple
  - `Double` ⇒ Precisión doble
- `Bool` ⇒ Es un valor Booleano
- `Char` ⇒ Es un caracter
- `Eq` ⇒ Tiene definida la igualdad
- `Ord` ⇒ Es ordenable
- `Enum` ⇒ Es enumerable
- `Show` ⇒ Se puede mostrar como texto
- `Read` ⇒ Se puede obtener a partir de texto

Operadores Básicos

- `+` ⇒ Suma
- `-` ⇒ Resta o cambio de signo
- `*` ⇒ Multiplicación
- `/` ⇒ División
- `div` ⇒ División entera
- `mod` ⇒ División modular
- `**` ⇒ Potencia con argumentos `Floating`
- `^` ⇒ Potencia con primer argumento `Num` y segundo `Integral`
- `%` ⇒ Simplifica la relación entre dos `Integral`

Operadores Básicos

- `==` $\Rightarrow$ Igual
- `/=` $\Rightarrow$ Distinto
- `<`, `<=` $\Rightarrow$ Menor, menor igual
- `>`, `>=` $\Rightarrow$ Mayor, mayor igual
- `&&` $\Rightarrow$ Y lógico
- `||` $\Rightarrow$ O lógico

Operadores y Llamado a Funciones

- Los operadores son funciones con *definición especial*
- Los operadores son *infijos*
- Se pueden cambiar a *prefijos* usando paréntesis
- Las funciones no requieren paréntesis
- Las funciones son *prefijas*
- Se pueden cambiar a *infijas* con comillas francesas (`)

Funciones sobre Datos

- `abs` $\Rightarrow$ ...

Listas y Cadenas

`:` $\Rightarrow$ Construcción de listas

`++` $\Rightarrow$ Concatenación de listas

`!!` $\Rightarrow$ Indexación de listas

`elem` $\Rightarrow$ El elemento pertenece

`notElem` $\Rightarrow$ El elemento no pertenece

Funciones sobre Listas

- `maximum` $\Rightarrow$ `...`