Compiladores

- Haskell -

3CM7

Vargas Romero Erick Efraín
Prof. Tecla Parra Roberto

Instituto Politécnico Nacional
Escuela Superior de Cómputo
Juan de Dios Bátiz, nueva industrial Vallejo
07738 ciudad de México

Chapter 1

Haskell

1.1 ¿Qué es Haskell?

Es un lenguaje de programación estandarizado multi-propósito puramente funcional con semánticas no estrictas y fuertemente tipado. Su nombre se debe al lógico estadounidense Haskell Curry, debido a su aportación al cálculo lambda, el cual tiene una gran influencia en el lenguaje. En Haskell, "una función es un ciudadano de primera clase" del lenguaje de programación. Como lenguaje de programación funcional, el constructor de controles primario es la función. El lenguaje tiene orígenes en las observaciones de Haskell Curry y sus descendientes intelectuales.

1.2 Historia

En 1985, se publica Miranda, los lenguajes funcionales proliferaron en este año. En 1987, existían compitiendo entre si más de una docena de lenguajes de programación puros funcionales no estrictos. Durante la conferencia de lenguajes de programación funcional y arquitecturas de ordenador en Portland, Oregón, se mantuvo un encuentro durante el cual se alcanzó un fuerte consenso entre sus participantes para formar un comité que definiese un estándar abierto para tales lenguajes. Esto se realizó con el propósito expreso de consolidar los lenguajes existentes en un único que sirviera como base para la investigación futura en diseño de lenguajes. La primera versión de Haskell, se definió en 1990. Los esfuerzos del comité resultaron en una serie de definiciones del lenguaje, que culminaron a finales de 1997 en Haskell 98, que se intentó fuera del lenguaje mínima, estable y portable, junto con una biblioteca estándar asociada para la enseñanza, y como base de futuras extensiones. El comité expresamente aprobó la creación de extensiones y variantes de Haskell 98 mediante la adición e incorporación de catacterísticas experimentales.

Para enero de 1999, el estandar del lenguaje Haskelll 98 se publicó en "The Haskell 98 Report". En enero de 2003, se publicó una versión revisada en "Haskell 98 Language and Libraries: The Revised Report". El lenguaje continua evolucionando rápidamente, con implementaciones y Hugs y de GHC, que representan el actual estandar de facto. A principios de 2006 comenzó el proceso de definición de un sucesor estandar de Haskell 98, llamado informalmente Haskell. En 2010 se lanza Haskell 2010

1.3 Tipos simples predefinidos

En Haskell, y en lo siguiente, "o::t" significa que el objeto "o" es un miembro del tipo "t" y "t -> s" es un tipo, específicamente una función, que consume algo del tipo "t" y produce algo de tipo "s", el operador "->" se nida por el derecho, ya que, "t -> s -> r" quiere decir "t -> (s -> r)"

1.3.1 El tipo Bool

Los valores con este tipo representan expresiones lógicas cuyo resultado puede ser true o false

Funciones y operadores

- (&&) :: Bool -> Bool -> Bool: Conjunción lógica
- (||) :: Bool -> Bool -> Bool: Disyunción lógica
- not :: Bool -> Bool: Negación lógica
- otherwise :: Bool: Función constante que devuelve true

1.3.2 Tipo Int

Los valores de este tipo son números enteros que van de$[-2^{29}, 2^{29} - 1]$

1.3.3 Tipo Integer

Enteros de precisión ilimitada que tienen las mismas funciones y operadores del tipo int

1.3.4 Tipo Float

Números reales

Funciones y operadores

- (+), (-), (*), (/), ($\hat{}$) :: Float -> Float -> Float: Suma, resta, multiplicación, división real y potencia de exponente entero.
- abs, signum, negate :: Int -> Int: Valor absoluto, signo y negación
- (**) :: Float -> Float: Potencia de exponente real

1.3.5 Tipo Double

Los valores son números reales, de mayor rango y con aproximaciones más precisas que los tipo Float.

1.3.6 Tipo Char

Valores de tipo caracter que se encuentran en una masa de alta complejidad de en una suma de caracteres dados con su alta definición. Antes de utilizar esta función en hugs debemos utilizar IMPORT CHAR antes de crear el algoritmo.

1.3.7 Tuplas

Los elementos que forman una tupla pueden ser del mismo tipo o de diferentes tipos. Es un conjunto de componentes relacionados.

1.3.8 Listas

Los valores de este tipo son una colección de elementos del mismo tipo. Existen dos constructores para listas:

1. '['Elementos separados por comas']' ejemplo: [1, 2, 3, 4, ... , n]
2. (primer elemento: resto de la lista) por ejemplo (1: (2: (3: (4: [ ]))))

1.4 Condicionales

Al igual que en otros lenguajes de programación, Haskell provee varias formas de definir funciones dentro de las cuales se puede elegir entre un cierto número de posibles resultados.

1.4.1 Operadores lógicos

- > mayor
- < menor
- >= mayor o igual
- <= menor o igual
- /= diferente de
- == igual

1.4.2 IF-THEN-ELSE

Una forma de implementar estructuras condicionales en Haskell es con la estructura if-then-else, cuya síntaxis es: if EXPRESIÓN LÓGICA then ACCIÓN EN CASO POSITIVO else ACCIÓN EN CASO NEGATIVO