

Linguagens Funcionais

Sérgio Queiroz de Medeiros
sergio@ufs.br

19 de abril de 2012

Linguagens Funcionais x Linguagens Imperativas

- ▶ Linguagens funcionais caracterizam-se pelo uso intenso de subrotinas
- ▶ Limites entre categorias de linguagens é um pouco nebuloso
 - ▶ É possível escrever em um estilo funcional em várias linguagens imperativas
 - ▶ Muitas linguagens funcionais incluem características imperativas (atribuição e iteração)

- ▶ Fazem uso intenso de listas
- ▶ Uso de recursão para execução repetitiva
- ▶ Tendência de gerar uma quantidade significativa de dados temporários
 - ▶ Coletor automático de lixo
- ▶ Exemplos: Lisp, ML, Miranda, Haskell, OCaml, F#

- ▶ Vários modelos teóricos de computação foram propostos nos anos 30 por Alan Turing, Alonzo Church, Stephen Kleene, entre outros.
 - ▶ Turing \Rightarrow Máquina de Turing
 - ▶ Church \Rightarrow Lambda cálculo
- ▶ Depois provou-se que todos esses modelos eram igualmente poderosos
 - ▶ Tese de Church-Turing: Toda ‘função que seria naturalmente considerada computável’ pode ser computada por uma Máquina de Turing.

- ▶ Baseado na noção de expressões parametrizadas
 - ▶ Letra λ é usada para introduzir parâmetros
- ▶ Lambda cálculo foi a inspiração para a programação funcional
 - ▶ A saída de um programa é uma função matemática das suas entradas

Linguagens Imperativas nem sempre possuem...

- ▶ Funções de primeira classe e funções de alta ordem
- ▶ Polimorfismo extensivo
- ▶ Um tipo lista e operadores sobre listas
- ▶ Recursão
- ▶ Coleta de lixo

- ▶ Lisp foi a primeira linguagem funcional e ainda é a linguagem funcional mais usada
- ▶ Características de Lisp
 - ▶ Homogeneidade de programas e dados: um programa em Lisp é ele próprio uma lista
 - ▶ Auto-definição: definição da semântica de Lisp através de um interpretador escrito em Lisp
 - ▶ Interação com o usuário

Programas como Listas

```
(define compose
  (lambda (f g)
    (lambda (x) (f (g x)))))
((compose car cdr) '(1 2 3))    --> 2
```

```
(define compose2
  (lambda (f g)
    (eval (list 'lambda '(x) (list f (list g 'x)))
          (scheme-report-environment 5))))
((compose car cdr) '(1 2 3))    --> 2
```


Composição de Funções em Haskell

- ▶ É possível compor funções através do operador `.`

`(.) :: (t -> u) -> (s -> t) -> (s -> u)`

`f . g = \x -> f(g(x))`

- ▶ Exemplo:

`not :: Bool -> Bool`

`odd :: Int -> Bool`

`even = not . odd`

Ordem de Avaliação

- ▶ Expressões podem ser avaliadas em mais de uma ordem
- ▶ Argumentos de uma função podem ser avaliados antes da função ser chamada (ordem aplicativa)
- ▶ Função pode ser chamada sem os seus argumentos terem sido avaliados (ordem normal)
- ▶ A maioria das linguagens imperativas (e algumas funcionais, como Lisp) usa ordem aplicativa

- ▶ `double (x) return x + x`
 - ▶ `double (3 * 4)`

- ▶ `switch`

```
switch (x a b c)
```

```
if x > 0
```

```
    return a
```

```
else if x = 0
```

```
    return b
```

```
else if x > 0
```

```
    return c
```

- ▶ `switch -1 (1 + 2) (2 + 3) (3 + 4)`

Avaliação Estrita x Avaliação Preguiçosa (*Strict Evaluation* x *Lazy Evaluation*)

- ▶ Ordem de avaliação pode afetar não somente a velocidade de execução, mas também a terminação de um programa
- ▶ Avaliação estrita: requer que todos os argumentos de uma função devem ser definidos
- ▶ Avaliação preguiçosa: avalia um argumento de uma função somente quando eles é necessário para uma computação
- ▶ Avaliação preguiçosa é útil para criar estruturas de dados infinitas, que são construídas sob demanda
- ▶ Miranda e Haskell usam avaliação preguiçosa

Funções de Alta Ordem

- ▶ `apply`
- ▶ `filter`
- ▶ `fold`
- ▶ `compose`

- ▶ Há idiomas de programação comuns baseados em atribuição
 - ▶ Entrada/Saída
 - ▶ Inicialização de estruturas complexas
 - ▶ Mutação *in-place*
- ▶ Atualmente Haskell é a linguagem funcional em torno da qual é feita a maioria das pesquisas
- ▶ Adoção de linguagens funcionais (especialmente as estaticamente tipadas) em aplicações comerciais
 - ▶ Surgimento de F#

- ▶ Programming Language Pragmatics (Michael Scott)
 - ▶ Capítulo 10
- ▶ Concepts of Programming Languages (Robert Sebesta)
 - ▶ Capítulo 15