Estruturas de Linguagem

Programação Funcional

Francisco Sant'Anna

francisco@ime.uerj.br

http://github.com/fsantanna-uerj/EDL

Programação Funcional

In computer science, **functional programming** is a programming paradigm—a style of building the structure and elements of computer programs—that treats computation as the evaluation of mathematical functions and avoids changing-state and mutable data. It is a declarative programming paradigm, which means programming is done with expressions[1] or declarations[2] instead of statements. In functional code, the output value of a function depends only on the arguments that are input to the function, so calling a function *f* twice with the same value for an argument *x* will produce the same result *f(x)* each time. Eliminating side effects, i.e. changes in state that do not depend on the function inputs, can make it much easier to understand and predict the behavior of a program, which is one of the key motivations for the development of functional programming.

In contrast, imperative programming changes state with commands in the source language, the most simple example being assignment. Imperative programming does have functions—not in the mathematical sense—but in the sense of subroutines. They can have side effects that may change the value of program state. Functions without return values therefore make sense. Because of this, they lack referential transparency, i.e. the same language expression can result in different values at different times depending on the state of the executing program.[3]

Imperativo vs Funcional

- Comando vs Expressão
- Sequência vs Dependência
- Atribuição vs Definição

Programação Funcional

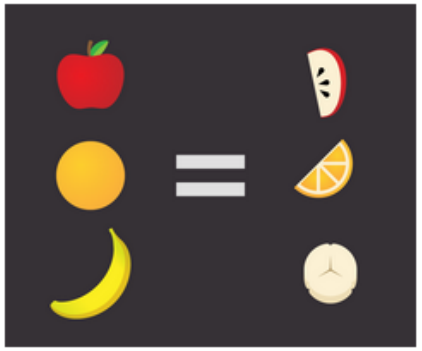
- Funções de Primeira Classe
  - podem ser criadas, passadas, retornadas
- Funções de Alta Ordem
  - recebem ou retornam funções
- Funções Puras
  - sem efeito colateral
  - qualquer ordem de avaliação (inclusive em paralelo)
- Funções Recursivas
  - auto referência

Programação Funcional 101

- listas
- map
- filter
- fold (reduce)

Map

You have an array of items and want to transform each of them. The result is a new array of the exact same length containing the manipulated items.



http://atendesigngroup.com/blog/

Listas

- Conjunto infinito de valores homogêneos
- Principal mecanismo estrutural em linguagens funcionais
- Imutáveis
- Operações (em Haskell)
  - Literais: `[1,2,3]`
  - Construção: `:` `// m = 0 : [1,2,3]`
  - Cabeça: `head` `// head m → 0`
  - Cauda: `tail` `// tail m → [1,2,3]`

Map

- Transformador de Listas
  - recebe uma lista `l` e uma função de transformação `f`
  - retorna uma nova lista `m` de mesmo tamanho com valores de tipos possivelmente diferentes
  - aplica `f` a cada elemento de `l`, *mapeando-os* para novos elementos em `m`
- Exemplos em Haskell

Filter

- Filtro de Listas
  - recebe uma lista `l` e uma função de filtro `f`
  - retorna uma nova lista `m` de no máximo mesmo tamanho com valores do mesmo tipo
  - aplica `f` a cada elemento de `l`, *filtrando* os que satisfizerem `f` para novos elementos em `m`
- Exemplos em Haskell

Fold

- "Agregador" de Listas
  - recebe uma lista `l`, um valor inicial v_0 e uma função de acumulação `f`
  - retorna um valor qualquer
  - aplica `f (`v_0`, `l_1`) ->`v_1

    `f (`v_1`, `l_2`) ->`v_2

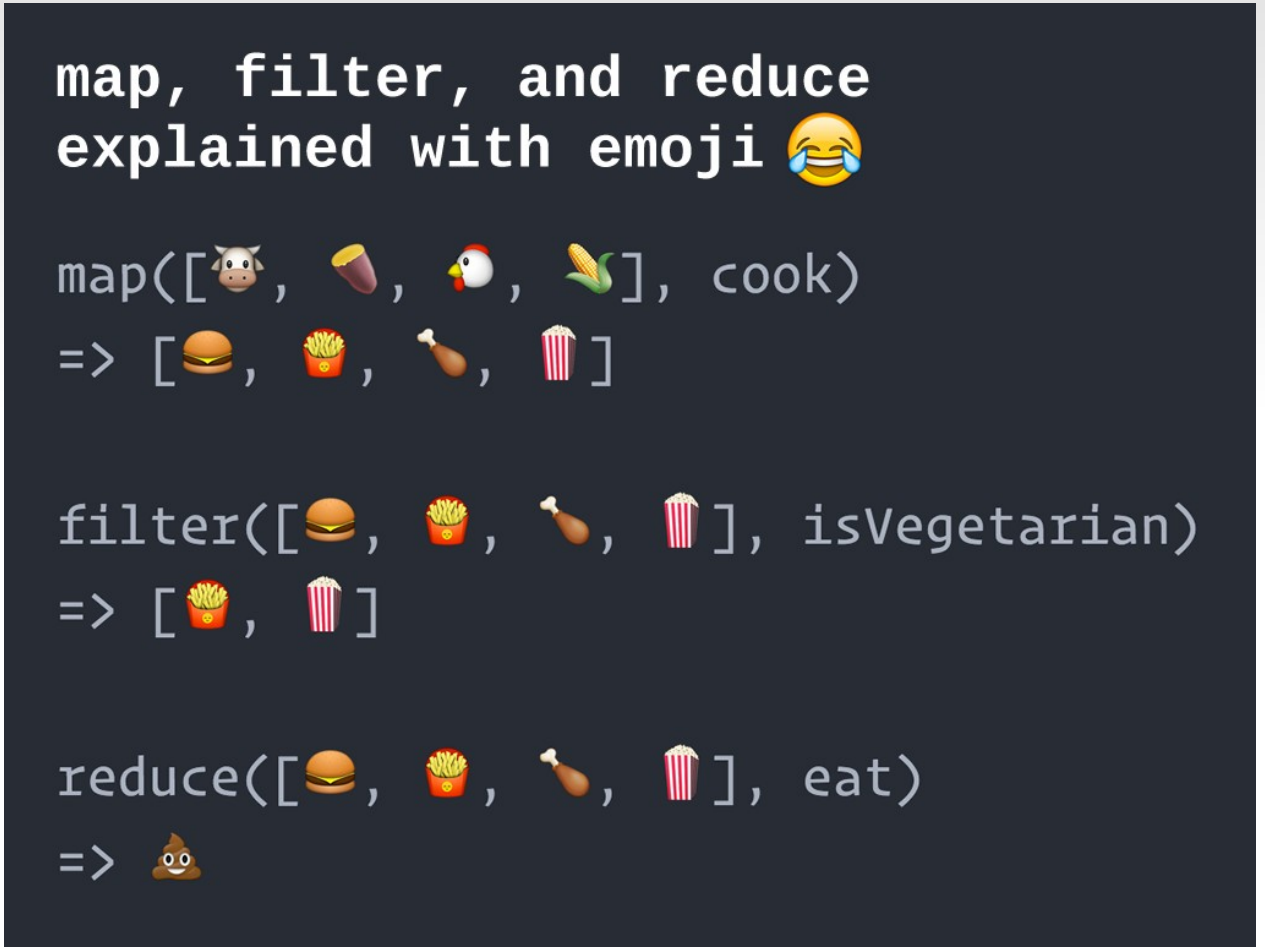
    `...`

    `f (`v_{n-1}`, `l_n`) ->`v_n (que é o valor final)

- Exemplos em Elm

Programação Funcional 101

- listas
- map
- filter
- fold (reduce)

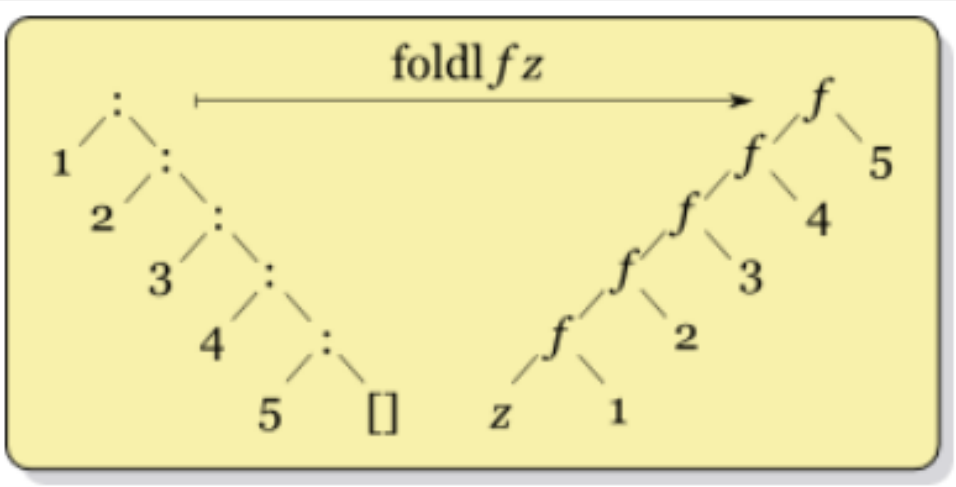


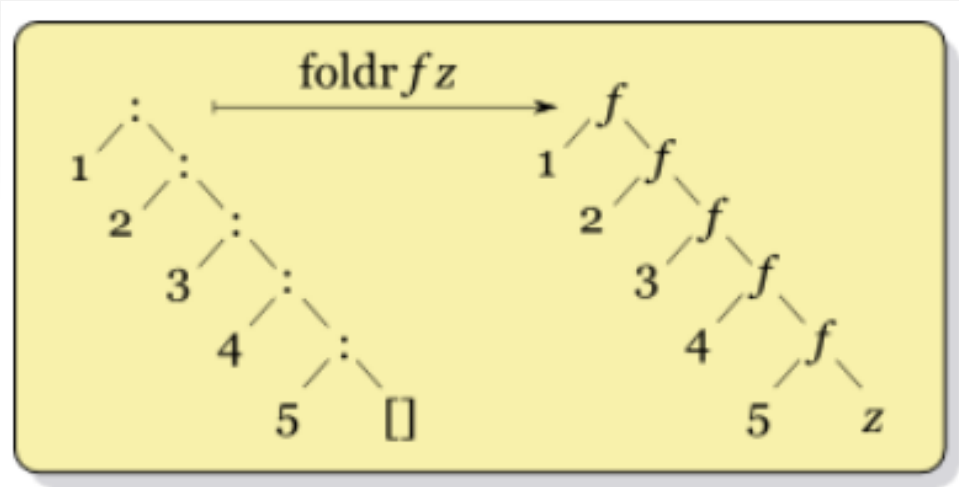
http://www.globalnerdy.com/2016/06/23/map-filter-and-reduce-explained-using-emoji/

Implementação

- listas
- map
- tipos polimórficos (tipos genéricos)
- filter
- fold (reduce)

foldl vs foldr





Imperativo vs Funcional

- Comando vs Expressão
- Sequência vs Dependência
- Atribuição vs Definição

Programação Funcional

- Funções de Primeira Classe
  - podem ser criadas, passadas, retornadas
- Funções de Alta Ordem
  - recebem ou retornam funções
- Funções Puras
  - sem efeito colateral
  - qualquer ordem de avaliação (inclusive em paralelo)
- Funções Recursivas
  - auto referência

Árvores

- Tipo
- Recursão