

PROGRAMAÇÃO FUNCIONAL

Antônio Victor Costa Passos
José Almeida Júnior
Raimundo Alan Freire Moreira
Otávio Rocha Neto

Tópicos

- × Introdução à Programação funcional
- × Programação funcional X Imperativa
- × Trabalhando com Programação funcional
- × Vantagens e desvantagens

DADOS HISTÓRICOS

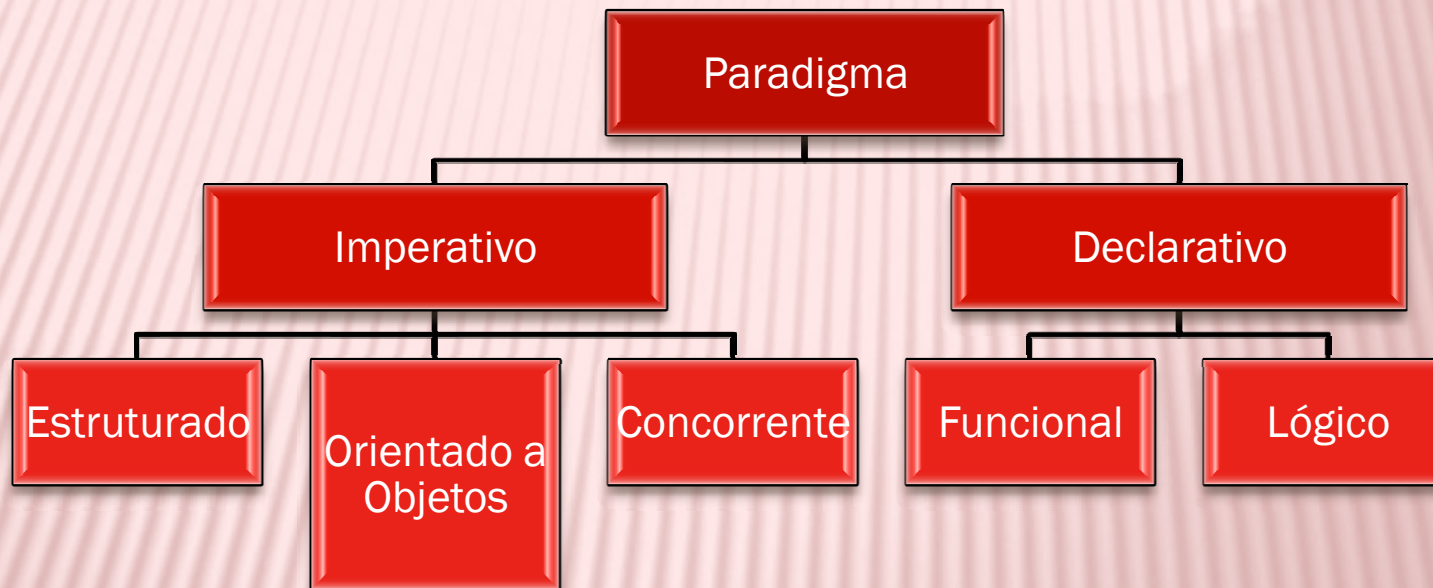
- Programação funcional surgiu para aproximar a computação da matemática pura.
 - Em 1930 Alonzo Church idealizou a primeira linguagem funcional, o calculo de lambda, porém nunca foi projetada para ser executada em um computador.
-

ALONZO CHURCH
CRIADOR DO CÁLCULO
DE LAMBDA



-
- × Anos 50 surge LISP, desenvolvida por Jhon McCarthy .
 - × Nos anos 1970 a linguagem ML foi criada pela Universidade de Edimburgo
 - × David Turner desenvolveu a linguagem Miranda na Universidade de Kent.
 - × A linguagem Haskell foi lançada no fim dos anos 1980 em uma tentativa de juntar muitas idéias na pesquisa de programação funcional.

PARADIGMAS



PARADIGMA FUNCIONAL

- Trata a computação como uma avaliação de funções matemáticas e que evita estado ou dados mutáveis.
 - $f(x) = x^2 + 2$
 - Linguagem funcional você pode passar funções como parâmetros para outras funções e devolvê-los como valores.
-

-
- ✘ Recursividade - As principais operações nesse tipo de programação são a composição de funções e a chamada recursiva de funções.

$$\sigma_n = f(\sigma_0)$$

PROGRAMAÇÃO FUNCIONAL X PROGRAMAÇÃO IMPERATIVA

- Alocação de variáveis
 - Declaração de variáveis
-

Soma 100 C

```
int total = 0;  
for (i = 1; i <= 100; i++)  
    total = total + 1;
```

Total será
alterada 100
vezes.
Efeito colateral.

Soma100 Haskell

```
s1_100:: integer  
s1_100 = sum [1..100]
```

S1_100 só será
alterada uma vez
durante todo o
programa.

- **COMANDOS DE INTERAÇÃO**

Fatorial C

```
int fatorial(int n)
{
    int r = 1;
    while(n > 1)
        r *= n--;
    return r;
}
```

Fatorial LISP

```
(defun fatorial (n)
  (if (= n 0) 1
      (* n (fatorial (- n 1)))))
```

TRABALHANDO COM LPS FUNCIONAIS

- ✗ avalia uma expressão e a reduz a uma resposta

```
> 5 * 3
```

```
15
```

- ✗ característica muito importante da programação funcional é a construção de *definições* ou *scripts*.

```
square x = x * x
```

```
> square (2 + 3)
```

```
25
```

APLICAÇÃO PRÁTICA

Fatorial Haskell

```
fatorial :: Int -> Int
fatorial 0 =
fatorial n = n * fatorial (n -
1)
```

Fibonacci Haskell

```
fib 0 = 0
Fib 1 = 1
fib n = fib (n - 2) + fib (n - 1)
```

LPS FUNCIONAIS - VANTAGENS E DESVANTAGENS

Vantagens:

- ✘ a codificação com menos potencial para erros;
- ✘ correção mais simples do que normalmente espelhar a estrutura do código.
- ✘ Programas mais curtos, mais fáceis de modificar
- ✘ Manipulação mais simples de programas, boa legibilidade

DESVANTAGENS:

- × Programas exigem sempre toneladas de memória.
- × Os programas funcionais tendem a correr lentamente
- × não interagem bem com outros programas
- × Não interagem com processos do sistema operacional