Revisão para a Prova 1

Programação Funcional

Prof. Maycon Amaro

Exercício 1

Mostre o passo-a-passo da avaliação do λ-termo abaixo até sua formal normal, utilizando uma das seguintes estratégias de avaliação: *normal-order*, *call-by-name*, *call-by-value*. Indique a estratégia utilizada.

$$(\lambda m.\lambda n.\lambda b.b\ m\ n)\ (\lambda t.\lambda f.f)\ (\lambda t.\lambda f.t)\ (\lambda t.\lambda f.f)$$

Exercício 2

Mostre o passo-a-passo da avaliação do λ-termo abaixo até sua formal normal, utilizando uma das seguintes estratégias de avaliação: *normal-order*, *call-by-name*, *call-by-value*. Indique a estratégia utilizada.

$$(\lambda m.\lambda n.m\ n\ m)\ (\lambda t.\lambda f.f)\ (\lambda t.\lambda f.t)$$

Exercício 3

Mostre o passo-a-passo da avaliação do λ-termo abaixo até sua formal normal, utilizando uma das seguintes estratégias de avaliação: *normal-order*, *call-by-name*, *call-by-value*. Indique a estratégia utilizada.

$$(\lambda m.\lambda n.\lambda s.\lambda z.m\ (n\ s)\ z)\ (\lambda s.\lambda z.z)\ (\lambda s.\lambda z.s\ z)$$

Exercício 4

Mostre o passo-a-passo da avaliação do λ-termo abaixo até sua formal normal, utilizando uma das seguintes estratégias de avaliação: *normal-order*, *call-by-name*, *call-by-value*. Indique a estratégia utilizada.

$$(\lambda m.\lambda n.m\ m\ n)\ (\lambda t.\lambda f.f)\ (\lambda t.\lambda f.t)$$

Exercício 5

Descreva com suas próprias palavras o conceito de **transparência referencial**. Pode utilizar exemplos de código em qualquer linguagem para ilustrar sua explicação.

Exercício 6

Implemente a função abaixo, que calcula o produto dos números pares presentes na lista informada. Utilize funções de ordem superior ao invés de recursão explícita. Exemplo: `prod_par [1, 2, 3, 4, 5]` retorna `8`.

```
prod_par :: [Int] -> Int
```

Exercício 7

Implemente a função abaixo, que determina qual o maior número na lista informada. Utilize recursão explícita ao invés de funções de ordem superior. Exemplo: `maior [7, 2, 6, 9, 4, 2]` retorna `9`. Para listas vazias, retorne 0.

```
maior :: [Int] -> Int
```

Exercício 8

Refaça o exercício anterior, utilizando funções de ordem superior ao invés de recursão explícita.

Exercício 9

Defina um tipo de dado algébrico `Vinculo` que corresponde à seguinte situação: um vínculo pode ser de aluno ou de funcionário. Alunos possuem um nome e um número de matrícula. Funcionários possuem um nome e a informação se são efetivos ou temporários.

```
data TipoFuncionario = Efetivo | Temporario
data Vinculo =
```

Exercício 10

Considerando o tipo de dado algébrico `Roupa`, implemente a função `troca` que troca as roupas das duas modelos informadas.

```
data Cor = Vermelho | Azul | Rosa | Branco
data Roupa = BlusaECalca Cor Cor | Vestido Cor
data Modelo = Modelo { nome :: String, idade :: Int, roupa :: Roupa }

troca :: (Modelo, Modelo) -> (Modelo, Modelo)
```