

Ficha 8

Semântica das Linguagens de Programação

2019/20

1. Considere a seguinte expressão A da linguagem de programação funcional estudada:

$$\text{let } f \equiv \lambda\langle x, y \rangle. x + y \text{ in } f \langle 5, 6 \rangle$$

- (a) Construa uma árvore de prova do juízo $\vdash A : \mathbf{Int}$.
- (b) Calcule o valor de A , usando a semântica de avaliação *call-by-value* da linguagem (deve começar por traduzir o açúcar sintático utilizado).

2. Considere a seguinte expressão **FACT**

$$\text{letrec fact} \equiv \lambda n. \text{if } n = 0 \text{ then } 1 \text{ else } n * \text{fact } (n - 1) \text{ in fact}$$

- (a) Construa uma árvore de prova do juízo $\vdash \mathbf{FACT} : \mathbf{Int} \rightarrow \mathbf{Int}$.
- (b) Prove que a avaliação CBV de $\text{letrec fact} \equiv \dots \text{ in } (\text{fact } 1)$ produz o valor 1.

3. Considere a seguinte expressão **APP**

$$\begin{aligned} \text{letrec append} &\equiv \lambda x. \lambda y. \text{listcase } x \text{ of } (y, \lambda h. \lambda t. h :: \text{append } t y) \\ &\text{in append } (1 :: \text{nil}) (2 :: 7 :: 8 :: \text{nil}) \end{aligned}$$

- (a) Construa uma árvore de prova do juízo $\vdash \mathbf{APP} : \mathbf{List Int}$
- (b) Apresente a avaliação CBV da expressão **APP** até à sua forma canónica.

4. Tendo definido funções de ordem superior, podemos definir novas funções utilizando definições não recursivas. Por exemplo:

$$\begin{aligned} \text{letrec foldr} &\equiv \lambda f. \lambda z. \lambda l. \text{listcase } l \text{ of } (z, \lambda h. \lambda t. f h (\text{foldr } f z t)) \text{ in} \\ \text{let append} &\equiv \lambda x. \lambda y. \text{foldr } (\lambda h. \lambda r. h :: r) y x \text{ in} \\ \text{append } (1 :: \text{nil}) &(2 :: 7 :: 8 :: \text{nil}) \end{aligned}$$

Apresente uma definição alternativa para a função **fact**.

5. Use a linguagem de programação funcional que estudou para definir as seguintes funções:

- (a) Valor absoluto de um inteiro.
- (b) Testar se uma lista de inteiros está ordenada.

6. Produza uma extensão da linguagem de programação funcional estudada, por forma a incluir um tipo de árvores binárias (com números inteiros nos nós intermédios e folhas vazias).

- (a) Defina sintaxe abstracta, regras de inferência de tipos, formas canónicas e regras de avaliação CBV apropriadas.
- (b) As árvores binárias podem ser vistas como estruturas de dados construídas à custa das alternativas e dos tuplos, vendo os seus construtores e eliminadores como açúcar sintático.

Apresente esta definição alternativa.