Interpretação e Compilação de Linguagens de Programação

Segundo Teste 2012-2013 — 17 de Maio de 2013

Notas: O teste pode ser realizado com consulta de materiais em suporte papel trazidos pelo aluno. O exame tem a duração de 1h30. Podem ser anexadas folhas extra ao teste se necessário.

Nome:_____ Número:_____

Q-1 [10 val.] Esta questão trata da análise da semântica de tipos linguagens de programação. Considere a linguagem *Vec*, cuja principal característica é a utilização de vetores de tamanho variável. A sintaxe concreta é definida pela seguinte gramática:

$$\begin{array}{rcl} E & ::= & num \mid E_1{+}E_2 \mid E_1/E_2 \mid E_1{=}E_2 \mid E_1 < E_2 \mid x \mid \mathsf{decl}\ x = E_1\ \mathsf{in}\ E_2 \\ & \mid & \mathsf{array}[E_1 \ldots E_2] \mid E_1[E_2] := E_3 \mid E_1[E_2] \mid E_1[E_2{...}E_3] \\ & \mid & \mathsf{ifnzero}\ E_1\ \mathsf{then}\ E_2\ \mathsf{else}\ E_3 \mid \mathsf{fun}\ x \to E \mid \mathsf{fun}\ x[z{...}y] \to E \mid E_1\ E_2 \end{array}$$

As construções básicas da linguagem *Vec* são **os números inteiros** (*num*) e as operações habituais sobre os números inteiros, aqui simbolizadas nas operações $E + E$, E/E, $E = E$ e $E < E$; as operações de comparação denotam zero no caso negativo e um valor diferente de zero no caso positivo; os **identificadores** (x); a **declaração de identificadores** $\mathsf{decl}\ x = E_1\ \mathsf{in}\ E_2$ com a semântica habitual; os **vetores** definidos pela expressão $\mathsf{array}[E_1 \ldots E_2]$, onde o valor inteiro denotado pela expressão E_1 e o valor inteiro denotado pela expressão E_2 representam o primeiro e último índices do vetor (inclusivé); ou por restrição de um vetor existente $E_1[E_2{...}E_3]$, que define uma referência para o vetor denotado pela expressão E_1 com os índices denotados pelas expressões E_2 e E_3. A linguagem *Vec* também inclui a operação básica de afetação de uma posição de um vetor $E_1[E_2] := E_3$ e consulta do valor contido por uma dessas posições $E_1[E_2]$. A expressão $\mathsf{ifnzero}\ E_1\ \mathsf{then}\ E_2\ \mathsf{else}\ E_3$ é condicional denotanto o valor da expressão E_2 se a expressão E_1 não denotar zero, e E_3 caso contrário. A linguagem inclui a abstração $\mathsf{fun}\ x \to E$ e a abstração com pattern matching para vetores $\mathsf{fun}\ x[z{...}y] \to E$. A linguagem *Vec* é avaliada segundo uma **estratégia de avaliação por valor** (*call-by-value*), os vetores são passados **por referência**, e resolução estática de nomes.

a) [3 val.] **Apresente** um exemplo de uma função de pesquisa binária num vetor ordenado utilizando a restrição de vetores como mecanismo base.

Número:_____

b) **[3 val.]** **Enumere** cuidadosamente os erros de execução em que programas da linguagem *Vec* podem incorrer.

c) **[2 val.]** **Defina** a linguagem de tipos necessária para tipificar a linguagem *Vec*. Pode usar classes Java (nomes das classes e listas de campos), ou um tipo de dados abstrato (por exemplo, em ML ou Haskell).

d) **[2 val.]** Indique as restrições que a linguagem de tipos impõe à linguagem *Vec*.

Número:_____

Q-2 **[10 val.]** Esta pergunta trata da definição da semântica de tipos de uma linguagem de programação tipificada, *TVec*, uma versão tipificada da linguagem *Vec*.

a) **[2 val.]** **Defina** a sintaxe abstrata das construções da linguagem *TVec* que precisarem de ser modificadas em relação à linguagem *Vec*.

b) **[6 val.]** **Defina** a semântica de tipos para as construções da linguagem *TVec* através da implementação do método **typecheck** de cada uma das classes correspondentes em Java, uma classe visitante ou através de uma função de tipificação recursiva em OCaml ou Haskell.

Número:_____

(continuação da alínea 1.c))

c) **[2 val.]** Enumere os erros de execução em que os programas bem tipificados da linguagem *Vec* podem incorrer.