

O *Environment* é um conjunto de associações entre um nome e um valor. Quando a aplicação se depara com um identificador ela deve procurá-lo no *Environment* para fazer a substituição. Do modo como está implementado, esta busca é linear. Portanto, uma mudança nessa estrutura de dados poderia otimizar o algoritmo.

## Conceitos de Linguagens de Programação

### Primeira Prova — 2017

- (2.0 pontos) O *Environment* que utilizamos é implementado como lista. O que muda, em termos de uso, se implementássemos como tabela de *hash*? E quanto ao *Storage*, haveria outra estrutura de dados interessante?
- (2.0 pontos) O que determina o tamanho do *Environment*? É possível determinar o seu tamanho máximo olhando apenas para a expressão a ser calculada? Explique.
- (3.0 pontos) Simule a seguinte ExprC

```
(appC (lamC 'x (appC (lamC 'y (plusC (idC 'x) (idC 'y)))
                    (plusC (idC 'z) (numC 28)))
      ) (numC 2))
```

*fun* (pointing to the inner lambda)  
*arg = 42* (under the argument 2)

quando o *Environment* contém

```
( (bind y (numV 9)) (bind z (numV 14)) mt-env )
```

*nome* *valor*      *nome* *valor* (above the bind pairs)

- (1.0 ponto) O que acontece se tivermos um identificador no valor do argumento, como neste caso:

```
(interpS '(call (func f (func x (call f 10))) (+ x y)))
```

*função* (above the first call)  
*arg* (above the argument y)

O que deveria acontecer e o que acontece? Por quê?

- (2.0 pontos) Se pensarmos em objetos como *closures*, podemos dizer que uma classe é uma função que prepara um *environment* adequado para as instâncias, retornadas por ela? Elabore.

*Sim.*

**Divirta-se e Boa Sorte!**

