# Paradigmas de Linguagens de Programação
# Exame Escrito

### Centro de Informática - UFPE
**16 de agosto de 2005**

**Questão 1 [1,5]** Defina uma função *proc* que recebe um elemento *p* a ser processado e uma lista de funções a serem aplicadas ao elemento, na ordem em que aparecem na lista. Considerando uma lista de funções *fs = [f1,...,fn]*, o resultado de *proc(p,fs)* deve ser *fn(...f2(f1(p)))*. Defina uma outra função *undo* que efetua o preocessamento inverso, assumindo a existência de um operador *inv* que, quando aplicado a uma função *f* resulta no processamento invserso ao efetuado por *f*. Portanto, *undo(p,fs)* deve resultar em *inv(f1)(inv(f2)...(inv(fn)(p)))*.
[**Sugestão**: Defina *undo* utilizando *proc*].

**Questão 2 [1,5]** Com relação à questão anterior, estruture a prova do seguinte teorema

*undo(proc(p,fs),fs) = p*

assumindo que *inv(f)(f(p)) = p*. Prove o caso base e explique os principais passos para provar o passo indutivo.

**Questão 3 [1,5]** Compare variáveis introduzidas por declarações com variáveis da *heap*, destacando como estas últimas são criadas e as vantagens e desvantagens de cada tipo de variável.

**Questão 4 [1,5]** Linguagens cujo projeto é uniforme tendem a oferecer maior poder de expressão e a ser mais facilmente aprendidas por programadores. Apresente exemplos de decisões de projeto de linguagens que limitam desnecessariamente certas construções e mencione possíveis soluções.

**Questão 5 [4,0]** Redefina a linguagem LF1 com o conceito de *script*, que permite ao programador introduzir declarações e avaliar expressões em qualquer ordem, sem a necessidade de utilizar expressões **let**. Um exemplo de script é apresentado a seguir:

```
fun suc x = x + 1
var y = 3
eval suc y
fun add x y = x + y
eval add(2,y)
```

Sua execução produz a saida 4, 5.

a) Defina a BNF para a linguagem LF1 redefinida.
b) Defina a interface do ambiente de execução com todos os métodos necessário à implementação de um interpretador para a linguagem redefinida. Explique suas decisões de projeto, mas NÃO precisa implementar os métodos.
c) Implemente as classes que representem um script, as declarações de funções e variáveis, e avaliação de expressões (cláusula Eval). Reutilize as classes já implementadas para LF1 sempre que possível e ignore verificação de tipos. Considere ainda o seguinte:
    I. O processamento de uma declaração, de função ou variável, com o mesmo nome de uma já processada resulta na substituição da existente pela nova.
    II. A cláusula **let** não faz parte da linguagem redefinida.

Apêndice 1. BNF de LF1.

Programa ::= Expressao

Expressao ::= Valor
| ExpUnaria
| ExpBinaria
| ExpDeclaracao
| Id
| Aplicacao
| IfThenElse

Valor ::= ValorConcreto

ValorConcreto ::= ValorInteiro | ValorBooleano | ValorString

ExpUnaria ::= "-" Expressao | "not" Expressao | "length" Expressao

ExpBinaria ::= Expressao "+" Expressao
| Expressao "-" Expressao
| Expressao "and" Expressao
| Expressao "or" Expressao
| Expressao "==" Expressao
| Expressao "++" Expressao

ExpDeclaracao ::= "let" DeclaracaoFuncional "in" Expressao

DeclaracaoFuncional ::= DecVariavel
| DecFuncao
| DeclaracaoFuncional "," DeclaracaoFuncional

DecVariavel ::= "var" Id "=" Expressao

DecFuncao ::= "fun" Id Id Id "=" Expressao

Aplicacao:= Id"("Expressao, Expressao")"

IfThenElse ::= "if" Expressao "then" Expressao "else" Expressao

## Classes Auxiliares

ValorFuncao
RestrictTypesVisitor
AmbienteExecucaoFuncional
ContextoExecucaoFuncional