Paradigmas de Linguagens de Programação
Exame Escrito

Centro de Informática - UFPE
13 de junho de 2008

Questão 1 [2,0] Defina uma função ***f*** que recebe duas listas como parâmetros: a primeira de pares do tipo ***X x Y*** e a segunda de elementos do tipo ***X***. Uma aplicação da função, como em ***f ps xs***, retorna uma lista com os pares de ***ps*** cujo primeiro elemento esteja na lista ***xs***. Por exemplo***, f [(x1,y1),(x2,y2),(x3,y3)] [(x1,x3)] = [(x1,y1),(x3,y3)]***

Questão 2 [2,0] Prove formalmente que ***f ps [] = []***

Questão 3 [2,0] Apresente argumentos contra e a favor com relação a Orientação a Objetos e Concorrência serem considerados paradigmas de programação.

Questão 4 [4,0] Estenda a Linguagem Imperativa 2 (veja BNF em anexo) permitindo a passagem de parâmetros por referência.

a) Defina a BNF para a linguagem redefinida, destacando apenas o que mudar.
b) Defina a interface do ambiente de execução com todos os métodos necessários à implementação de um interpretador para a linguagem redefinida. Explique suas decisões de projeto e os atributos necessários à implementação da interface do ambiente, mas NÃO precisa implementar os métodos.
c) Implemente as classes relacionadas à declaração e chamada de procedimento, esta última com o método executar (não precisa implementar o método que faz verificação de tipos).

Apêndice 1. BNF de LI2.

Programa ::= Comando

Comando ::= Atribuicao | ComandoDeclaracao
| While | IfThenElse
| IO | Comando ";" Comando
| Skip | ChamadaProcedimento

Atribuicao ::= Id ":=" Expressao
Expressao ::= Valor | ExpUnaria | ExpBinaria | Id
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

ComandoDeclaracao :: = "{" Declaracao ";" Comando "}"

Declaracao ::= DeclaracaoVariavel | DeclaracaoProcedimento
| DeclaracaoComposta
DeclaracaoVariavel ::= "var" Id "=" Expressao
DeclaracaoComposta ::= Declaracao "," Declaracao
DeclaracaoProcedimento ::= "proc" Id "(" [ListaDeclaracaoParametro] ")" "{" Comando "}"
ListaDeclaracaoParametro ::= Tipo Id | Tipo Id "," ListaDeclaracaoParametro

Tipo ::= "string" | "int" | "boolean"

While ::= "while" Expressao "do" Comando
IfThenElse ::= "if" Expressao "then" Comando "else" Comando
IO ::= "write" "(" Expressao ")" | "read" "(" Id ")"
ChamadaProcedimento ::= "call" Id "(" [ListaExpressao] ")"
ListaExpressao ::= Expressao | Expressao, ListaExpressao

Classes Auxiliares

AmbienteCompilacaoImperativa2

AmbienteExecucaoImperativa2
ContextoCompilacaoImperativa2
ContextoExecucaoImperativa2
ListaValor
Procedimento
DefProcedimento
ProcedimentoJaDeclaradoException
ProcedimentoNaoDeclaradoException