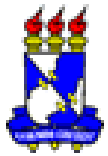
Framework para uma Família de LP

Prof. Alberto Costa Neto
alberto@ufs.br

Linguagens de Programação



Departamento de Computação
Universidade Federal de Sergipe

Linguagens

- Linguagem de Expressões 1 (LE1)
- Linguagem de Expressões 2 (LE2)
- Linguagem Funcional 1 (LF1)
- Linguagem Funcional 2 (LF2)
- Linguagem Funcional 3 (LF3)
- Linguagem Imperativa 1 (LI1)
- Linguagem Imperativa 2 (LI2)
- Linguagem Orientada a Objetos 1 (LO1)

LO1

Linguagem de Objetos 1

(Classes)

LO1 - Características

- Estende a LI1 com com **declarações de classes**, **criação dinâmica de objetos**, e **chamada de métodos**
- Ao invés do estado monolítico (PI), cada objeto tem o **status de uma entidade independente**, com **estado próprio** e **ações** que podem manipulá-lo
- Um **programa** é **comando precedido por declarações de classes**

LO1 – Características (cont.)

- A principal construção nova é a declaração de classes
- Atributos são similares às variáveis das LI1
- Métodos são similares aos procedimentos de LI2
  - Podem ser recursivos e parametrizados
  - Só podem ocorrer dentro de declarações de classes
  - Não têm status de valor

LO1 – Características (cont.)

- Na atribuição a um atributo, este deve ser qualificado (precedido do nome da instância)
- O acesso a um atributo pode ocorrer em expressões
- this representa o objeto corrente
- ValorNull é o valor de uma variável do tipo classe não inicializada
- A declaração de variável passa a requerer um tipo

Instanciação de Classes

- Objetos são criados a partir da construção new seguido do nome da classe
- A criação de objetos não é uma expressão na linguagem:
  - a construção é limitada à inicialização de uma declaração ou
  - ao comando (New) que define uma forma particular de atribuição para a criação de objetos

Declaração de Classes

- A declaração de uma classe envolve:
  - o nome da classe
  - a declaração dos atributos e dos métodos (sintaxe de procedimentos na LI2)

Ambiente de Execução

- É bem mais complexo do que o de LI2 e inclui na interface do ambiente inclui métodos para:
  - mapear nomes de classes em definições de classes
  - retornar a definição de uma classe dado seu identificador
  - mapear identificadores, incluindo this, em valores, incluindo referências (este método é herdado e reutilizado da interface Ambiente)
  - mapear referências em objetos
  - retornar um objeto associado a uma referência
  - retornar a referência para a próxima célula a ser alocada

Ambiente de Execução

```
public interface AmbienteExecucaoOO1
        extends Ambiente<Valor>{
   public void mapDefClasse(Id idArg, DefClasse defClasse) ;
   public DefClasse getDefClasse(Id idArg);
   public void mapObjeto(ValorRef valorRef, Objeto objeto);
   public Objeto getObjeto(ValorRef valorRef);
   public ValorRef getProxRef();
}
```

LO1 - Sintaxe

Programa ::= "{" DecClasse ";" Comando "}"

Comando ::= “skip” | Atribuicao | IO | Comando ";" Comando
| IfThenElse | While | ComandoDeclaracao
| New | ChamadaMetodo

Atribuicao ::= LeftExpression ":=" Expressao

LeftExpression ::= Id | AcessoAtributo

AcessoAtributo ::= LeftExpression.Id | this.Id

Expressao ::= Valor | ExpUnaria | ExpBinaria
| LeftExpression | this

Valor ::= ValorConcreto

ValorConcreto ::= ValorInteiro | ValorBooleano | ValorString
| ValorNull

LO1 – Sintaxe (cont.)

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

DecVariavel ::= Tipo Id "=" Expressao
| DecVariavel "," DecVariavel
| Tipo Id ":=" "new" Id

New ::= LeftExpression ":=" "new" Id

ChamadaMetodo ::= Expressao "." Id "(" ListaExpressao ")"
| Expressao "." Id "("")"

DecClasse ::= "classe" Id "{" DecVariavel ";" DecProcedimento "}"
| DecClasse "," DecClasse

Tipo ::= TipoClasse | TipoPrimitivo

TipoClasse ::= Id

LO1 - Exemplos

```
{ classe LValor {
   int valor = -100,    LValor prox = null;
   proc insere(int v) {
      if (this.valor == -100) then
         { this.valor := v; this.prox := new LValor }
     else {this.prox.insere(v)}
   },
   proc print() {  write(this.valor);
      if (not(this.prox == null)) then {this.prox.print()} else {skip}
   }
}; { LValor lv := new LValor; lv.insere(3); lv.insere(4); lv.print() }  }
```