Linguagens de Programação

UNIVERSIDADE FEDERAL FLUMINENSE
PREFEITURA MUNICIPAL DE RIO DAS OSTRAS

Expressões

Carlos Bazilio
carlosbazilio@id.uff.br
http://www.ic.uff.br/~bazilio/cursos/lp

Expressões

- São compostas de elementos atômicos (constantes ou variáveis) ou a combinação destes no uso de operações
  - a, “Alo mundo!”, 10.5
  - f(a, b, c), 10 / 7, g(“abc”, 4 * f(1, 2, 3))
- As operações podem ter notação infixa (a+b), pré-fixa (+(a,b)) ou pós-fixa ((a,b)+)
- Muitas linguagens utilizam a notação infixa por parecer mais natural; Entretanto, linguagens baseadas em LISP utilizam notação pré-fixada:
  - (* (+ 1 3) 2) // significa (1 + 3) * 2

Expressões

- Também há situações em que a operação infixa ocorre de forma múltipla
  - a = b != 0 ? a/b : 0;
- Apesar da notação infixa simples parecer ser mais natural, a usual definição de funções implica no uso de forma pré-fixada (i.e., em C)
- Algumas linguagens permitem a definição de operadores infixos (ML, R, Smalltalk, C++, ..)
- A notação pós-fixa é comum em algumas calculadoras, na linguagem Forth e em contruções de linguagens comuns (x++, x--, ..)

Expressões
Precedência

- Suponha a expressão em Fortran:

  a + b * c ** d ** e / f

- Como esta deve ser interpretada?

  ((((a + b) * c) ** d) ** e) / f *ou*

  a + (((b * c) ** d) ** (e / f)) *ou*

  a + ((b * (c ** (d ** e))) / f)

- A opção equivalente à primeira é a última!
- Isto se dá de acordo com as regras de precedência entre os operadores da linguagem

Expressões
Precedência

- Um exemplo de confusão na definição de precedências ocorre em Pascal

  if A < B and C < D then (* bla bla bla *)

- A expressão avaliada por Pascal é:

  A < (B and C) < D

- Ou seja, Pascal define maior precedência para operadores lógicos que os relacionais
- Linguagens como Smalltalk e APL não definem nenhum tipo de precedência: neste caso, os parênteses são o recurso utilizado para tal

Expressões
Associatividade

- Associatividade define se as expressões serão avaliadas da esquerda para a direita, ou o contrário
- A expressão 9 – 3 – 2 vale:
  - 9 – 3 – 2 = 6 – 2 = 4 (esquerda para a direita)
  - 9 – 3 – 2 = 9 – 1 = 8 (direita para a esquerda)
- Em Fortran, a expressão 4 ** 3 ** 2 vale 262144 (4 ** 9), e não 4096 (64 ** 2)
- Para efeitos de legibilidade e garantia do significado, o recomendado é sempre usar os ()

Expressões
Atribuições

- São o *bloco básico* das linguagens imperativas
- Sua sintaxe geral é:

lado_esquerdo = lado_direito

- Tipicamente, o *lado_esquerdo* refere-se a uma área de memória, enquanto que *lado_direito* refere-se ao conteúdo de uma área de memória

a = b

- Quando nos referimos a uma área de memória numa atribuição, costumamos dizer que a variável é um *l-value*, enquanto o conteúdo um *r-value*

Expressões
Atribuições

- Esta distinção entre *l-values* e *r-values* nem sempre é tão clara

```
(f(a)+3)->b[c] = 2;
```

- A atribuição acima é possível em C?
- Algumas linguagens (Algol, CLU, Lisp, Haskell, …) fazem distinção explícita empregando um *modelo de referência*
- Nestas linguagens, uma variável **não** é um nome que representa um valor numa área de memória, mas uma **referência** para o valor

Expressões
Atribuições

```
b := 2;
c := b;
a := b + c;
```

Expressões
Atribuições

- Linguagens como Clu, ML, Perl, Lua, Python e Ruby permitem múltiplas atribuições

```
a, b = c, d;
```

- Isto permite construções interessantes como

```
a, b = b, a;
```

que troca os valores entre a e b

- Esta característica ainda permite usos mais elegantes como funções que retornam mais de 1 valor:

```
a, b, c = func(d, e);
```

Expressões
Avaliação Curto-Circuito

- Usualmente, em expressões como

(a < b) and (b < c)

onde a >= b, a expressão (b < c) não chega a ser avaliada

- Linguagens que realizam esta otimização avaliam expressões por curto-circuito
- Este não é o caso de Pascal:

```
p = lista;
while (p <> nil) and (pˆ.key <> val) do p = pˆ.next;
```

- O operador and é o último a ser avaliado

Expressões
Avaliação Curto-Circuito

- Avaliações curto-circuito são úteis para se evitar diversas situações errôneas:
  - Acesso a vetores fora dos seus limites
  - Divisão por zero
  - Economia de tempo:

```
if (condicao_improvavel && funcao_custosa()) ...
```