

Pilhas e Filas – aplicações

Avaliação das expressões:

Cientistas de computação enfrentaram várias dificuldades ao conceberem a idéia de linguagens de programação de nível mais alto. Uma das maiores foi a questão como gerar instruções na linguagem de máquina, que poderiam avaliar qualquer expressão aritmética, por exemplo:

$$X \leftarrow A / B ** C + D * E - A * C$$

A expressão é formada por operadores (divisão, potência, soma, etc) e operandos (A, B, C, etc).

O primeiro problema com a compreensão do significado de uma expressão está em decidir em que ordem as operações devem ser executadas. Isso significa que cada linguagem deve definir singularmente essa tal ordem. Por exemplo, se A=4, B=C=2, D=E=3:

$$\begin{aligned} X &= 4 / (2 ** 2) + (3 * 3) - (4 * 2) \\ &= (4 / 4) + 9 + 8 \\ &= 2 \end{aligned}$$

Porém, a verdadeira intenção do programador poderia ter sido atribuir o valor ao X

$$\begin{aligned} &= (4 / 2) ** (2 + 3) * (3 - 4) * 2 \\ &= (4 / 2) ** 5 * (-1) * 2 \\ &= 32 * (-2) \\ &= -64 \end{aligned}$$

Naturalmente, ele poderia especificar esta última ordem de avaliação usando os parênteses

$$X \leftarrow (((A / B) ** (C + D)) * (E - A)) * C$$

Além dos parênteses, utilizamos uma tabela de prioridades, tal como a utilizada na PL/1:

Operador	Prioridade
**, unário -, unário +	6
*, /	5
+, -	4
<, >, =, etc	3
and	2
or	1

Como pode um compilador aceitar uma expressão assim e produzir um código correto? Reconstruindo a expressão em uma forma que chamaremos pós-fixada. A maneira habitual de se escrever expressões é conhecida como infixa – os operadores encontram-se entre os operandos. Na pós-fixada, os operadores aparecem depois dos operandos

INFIXO: $A * B / C$ PÓS-FIXO: $AB * C /$

O exemplo anterior seria:

INFIXO: $X \leftarrow A / B ** C + D * E - A * C$

PÓS-FIXO: $X \leftarrow ABC ** / DE * + AC * -$

Cada vez que computamos um valor, vamos armazená-lo em localização temporária T_i , $i \geq 1$. Lendo da esquerda para a direita, a primeira operação é potenciação:

Operação	Pós-fixada
$T_1 \leftarrow B ** C$	$AT_1 / DE * + AC * -$
$T_2 \leftarrow A / T_1$	$T_2 DE * + AC * -$
$T_3 \leftarrow D * E$	$T_2 T_3 + AC * -$
$T_4 \leftarrow T_2 + T_3$	$T_4 AC * -$
$T_5 \leftarrow A * C$	$T_4 T_5 -$
$T_6 \leftarrow T_4 - T_5$	T_6

Vantagens da notação pós-fixa:

- Elimina-se a necessidade dos parênteses
- Expressão pode ser avaliada fazendo o exame da esquerda para a direita, empilhando os operandos e analisando os operadores.
- Utilizam-se os operandos como números da pilha e, finalmente, também coloca-se o resultado dentro da pilha.

Algoritmo para conversão de Infixo para Pós-fixado:

- 1) coloque todos os parênteses na expressão;
- 2) mova todos os operadores para que eles substituam seus parêntes correspondentes do lado direito;
- 3) suprima todos os parêntes.

Por exemplo:

$$A / B ** C + D * E - A * C$$

colocando os parênteses:

$$(((A/(B**C))+(D*E))-(A*C))$$

$$ABC ** / DE * + AC * -$$

Problema do algoritmo: necessidade de dois passos – primeiro, a leitura da expressão e inserção dos parênteses e segundo, a movimentação efetiva dos operadores.

Podemos formar pós-fixado passando imediatamente todo operando para a saída.

A solução será armazená-los em uma pilha até um momento oportuno, quando serão passados para a direita.

Ex:

Infixo: $A + B * C$

Pós Fixo: $ABC * +$

Próxima Marca	Pilha	Saída
Nenhuma	Vazia	Nenhuma
A	Vazia	A
+	+	AB
B	+	AB

É necessário determinar se * será colocado no topo da pilha ou + removido.

Como * tem prioridade mais alta, devemos empilhar * produzindo

*	+ *	AB
C	+ *	ABC

Uma vez terminada a expressão de entrada, provocamos a saída de todos os operadores restantes na pilha e

$ABC * +$

Exemplo:

Infixo: $A * (B + C) * D$

Pós-fixado: $ABC + * D *$

Próxima Marca	Pilha	Saída
Nenhuma	Vazia	Nenhuma
A	Vazia	A
*	*	A
(*(A
B	*(AB
+	*(+	AB
C	*(+	ABC

Neste ponto queremos desmembrar a pilha até o parêntese esquerdo e então remover os parênteses esquerdo e direito; isso produzirá:

)	*	ABC+
*	*	ABC+*
D	*	ABC+*D
Feito	Vazio	ABC+*D*

Símbolo	Prioridade dentro da Pilha	Prioridade de Chegada
)	-	-
**	3	4
*, /	2	2
Binário +, -	1	1
(0	4

A regra será que os operadores estão sendo retirados da pilha até que a sua prioridade da pilha permaneça maior do que ou igual a prioridade entrante do novo operador.