UniAcademia Academia
Cursos de Bacharelado em Sistemas de Informação e Engenharia de Software
Exercício prático de Pilha/Fila
Professor. Luiz Thadeu Grizendi

Instruções de máquina

Como gerar instruções de máquina para avaliar corretamente uma expressão aritmética genérica?

Este é um problema que se apresentou como um dos desafios para os primeiros projetistas de linguagens de programação de alto nível. Uma expressão E como:

A × C / D + E / B - A × D

pode ter significados diferentes, dependendo das prioridades dos operandos, regras de avaliação ou uso de parênteses. Este problema já foi resolvido e uma solução simples para ele é discutida a seguir.

Uma expressão é composta por operandos e operadores. A expressão E contém 5 operandos: A, B, C, D e E; e os operadores ×, / , + e -.

O primeiro problema que deve ser resolvido para se entender o significado de uma expressão é decidir em qual ordem as operações devem ser executadas, o que depende do uso de parênteses e das prioridades de cada operador. Operadores com prioridade mais alta são avaliados primeiros. Um possível assinalamento de prioridades é:

Operador	Prioridade
× , /	2
+ , -	1

O segundo problema é como tratar uma expressão com dois operadores adjacentes de mesma prioridade (por exemplo: A × C / D) . Neste caso, uma regra adicional que diz que a avaliação deve ser feita da esquerda para a direita resolve a ambiguidade.

Definidas as prioridades e regras de avaliação, sabemos que a expressão

E = A × C / D + E / B - A × D

será avaliada como

(((A × C) / D) + (E / B)) - (A × D)

UniAcademia Academia
Cursos de Bacharelado em Sistemas de Informação e Engenharia de Software
Exercício prático de Pilha/Fila
Professor. Luiz Thadeu Grizendi

Como o compilador aceita expressões como "E" e, produz código de máquina correto?

A resposta é reorganizar a expressão original numa forma que é conhecida como *notação pós-fixada* ou notação polonesa (em homenagem ao matemático polonês Jan Lukasiewicz, que inventou a notação). Expressões escritas da maneira convencional (como E acima) estão em notação *infixada*, pois operadores aparecem entre ("in") os operandos.

Na notação pós-fixada, cada operador aparece *após* seus operandos. Por exemplo:

Infixada	**Pós-fixada**
A × B / C	A B × C /
A / B × C + D × E – A × C	A B / C × D E × + A C × -
(A / B) × (C + D) × (E - A) × C	A B / C D + × E A – × C × A

A notação pós-fixada tem as seguintes vantagens sobre a convencional: não há necessidade de parênteses, não há necessidade de se estabelecer as prioridades dos operadores, e a avaliação é muito mais simples.

Para se avaliar uma expressão na notação pós-fixada, basta percorrer a expressão da esquerda para a direita da seguinte maneira:

a) se encontrar um operando, empilhe-o
b) se encontrar um operador, execute a operação sobre os operandos que estiverem no topo da pilha e empilhe o resultado.

Ao final, o resultado da expressão estará no topo da Pilha. Em pseudo-linguagem, seria algo como:

```
result (F: Expressão Pós-fixada do tipo fila):
{
        S: Pilha;  // pilha de operandos
        enquanto a fila F não estiver vazia faça:
                x = dequeue(F);  // exclui o elemento do início da fila
                se x é operando então
                        empilhe(x, S);
                caso contrário,
                        op2= desempilhe(S)
                        op1= desempilhe(S)
                        resultado = op1 ? op2 ( executa a operação indicada por x = ?)
                        empilhe(resultado,S);
        end;
        retorne desempilhe(S)
}
```

UniAcademia Academia
Cursos de Bacharelado em Sistemas de Informação e
Engenharia de Software
Exercício prático de Pilha/Fila
Professor. Luiz Thadeu Grizendi

O problema de realizar a *tradução de uma expressão na notação infixada para a pós-fixada* é uma rotina cuja implementação é extremamente mais fácil utilizando a estrutura árvore, que será discutida em um outro capítulo.

Utilize a estrutura fila e pilha na implementação!

Exemplos de Entrada/Saída

A ativação da função que calcula o valor da expressão na forma pós-fixada, considerando, como entrada uma fila de operandos/operadores, que são do tipo string.

```
# expressão infixada 3 + 5 * 8
# expressão pósfixada 3 5 8 * +
# resultado 43
print(result(Queue(['3','5','8','*','+'])))
# expressão infixada 3 * 5 + 8
# expressão pósfixada 3 5 * 8 +
# resultado 23
print(result(Queue(['3','5','*','8','+'])))
```

Exemplos de Saída

```
43
23
```

BOM Trabalho!!!!