

微视频

表达式求值

中国海洋大学信息学院

魏振钢

Tel:0532-66781226

Email:wzgwzq@ouc.edu.cn

3.2 栈的应用举例： 表达式求值

限于二元运算符的表达式定义:

表达式 ::= (操作数) + (运算符) + (操作数)

操作数 ::= 简单变量 | 表达式

简单变量 ::= 标识符 | 无符号整数

在表达式求值中，一种简单直观、广泛使用的算法，通常称为“算符优先法”。

例如：对表达式 $4+2*3-10/5$ 求值，按“算符优先法”对其求解过程如下：

$$4+2*3-10/5=4+6-10/5=10-10/5=10-2=8$$

- 对于一般表达式如何求其值？

例如：

$$\text{Exp} = a \times b + (c - d / e) \times f$$



1. 数字的计算顺序如何？
2. 运算符的计算顺序如何？
3. 数字栈的入栈和退栈的条件如何？
4. 符号栈的入栈和退栈的条件如何？

任何一个表达式都是由操作数 (operand)、运算符 (operator) 和界限符 (delimiter) 组成的，我们称之为单词。

把运算符和界限符统称为算符，它们构成的集合命名为 OP。根据优先运算规则，在运算的每一步中，任意两个相继出现的算符 θ_1 和 θ_2 之间的优先关系至多是下面三种关系之一：

$\theta_1 < \theta_2$ θ_1 的优先权低于 θ_2

$\theta_1 = \theta_2$ θ_1 的优先权等于 θ_2

$\theta_1 > \theta_2$ θ_1 的优先权高于 θ_2

表3. 1各种基本算符间的优先关系

$\theta_1 \backslash \theta_2$	+	-	*	/	()	#
+	>	>	<	<	<	>	>
-	>	>	<	<	<	>	>
*	>	>	>	>	<	>	>
/	>	>	>	>	<	>	>
(<	<	<	<	<	=	
)	>	>	>	>		>	>
#	<	<	<	<	<		=

算符优先算法求表达式求值过程为：

- 1) 设立操作符栈**OPTR**，操作数或运算结果栈**OPND**；
- 2) 设表达式的结束符为“#”，预设运算符栈的栈底为“#”，操作数栈为空栈；
- 3) 依次读入表达式中每个字符，若是**操作数**则进**OPND**栈，若是**运算符**则和**OPTR**栈的栈顶运算符比较优先权后作相应操作，直至整个表达式求值完毕（即OPTR栈的栈顶元素和当前**读入的字符**均为‘#’）。

```
OperandType EvaluateExpression() {  
    //算术表达式求值的算符优先算法。设OPTR和OPND  
    //分别为运算符栈和运算数栈，OP为运算符集合。  
    InitStack (OPTR); Push(OPTR, ' #' );  
    initStack (OPND); c=getchar();  
    while (c!= ' #' || GetTop(OPTR)!= ' #' ) {  
        if (!In (c, OP)) { Push (OPND, c); c= getchar();}  
                                //不是运算符则进栈  
        else  
            switch (Precede( GetTop (OPTR), c)){  
                case ' < ' :                //栈顶元素优先权低  
                    Push (OPTR, c); c= getchar();  
                    break;
```

```
    case' =' : //脱括号并接受下一字符
        Pop (OPTR, c); c= getchar();
        break;
    case' >' : //退栈并将运算结果入栈
        Pop (OPTR, theta);
        Pop (OPND, b); Pop (OPND, a);
        Push (OPND, Operate (a, theta, b));
        break;
} //switch
} //while
return GetTop(OPND);
} //EvaluateExpression
```

算法 3.4

动画演示

Exp = 2 × 3 + (6 − 8 / 4) × 2#

OPND

14

OPTR

.
.
.
#