

NOTAÇÃO INFIXA E NOTAÇÃO POSFIXA

CONVERSÃO DE INFIXA PARA POSFIXA. Vamos considerar uma expressão infixa como uma seqüência de caracteres que pode conter somente as letras maiúsculas do alfabeto, os operadores aritméticos '+', '-', '*', '/', parênteses (esquerdo e direito) e termina com o caractere ponto-e-vírgula ';'. Vamos supor que a expressão infixa seja bem formada, isto é, não contém erros de sintaxe nem de posicionamento.

O processo de conversão de uma expressão infixa para a notação posfixa faz uso de uma pilha de caracteres, que vamos identificar por pilhaC. A expressão infixa é armazenada em uma lista de caracteres, que vamos identificar por expInfixa. A expressão posfixa, resultante do processo, é uma lista de caracteres e será identificada por expPosfixa.

O processo de conversão devolve a expressão expPosfixa com o caractere ponto-e-vírgula como símbolo terminal. Caso a expressão infixa seja nula (ou inválida), será devolvida a lista posfixa com um único caractere – o símbolo terminal. A tabela a seguir resume o procedimento de conversão:

Tabela de conversão

caractere infixa = ch	Topo da pilha = chp			
	∅	(+, -	*, /
Ident	colocar ch na posfixa	colocar ch na posfixa	colocar ch na posfixa	colocar ch na posfixa
(push	push	push	push
+, -	push	push	colocar chp na posfixa; pop	colocar chp na posfixa; pop
*, /	push	push	push	colocar chp na posfixa; pop
;	fim	erro	colocar chp na posfixa; pop	colocar chp na posfixa; pop
)	erro	pop	colocar chp na posfixa; pop	colocar chp na posfixa; pop

Esboço do processo de conversão

para cada caractere ch da expressão Infixa repita

caso ch seja

'(' : colocar ch na pilhaC;

'+', '-' : chp ← topo da pilhaC;

proceder conforme linha 3 da tabela de decisão *até que* push

'*', '/' : chp ← topo da pilhaC;

proceder conforme linha 4 da tabela de decisão *até que* push

'/' : chp ← topo da pilhaC;

proceder conforme linha 5 da tabela de decisão *até que* fim

')' : chp ← topo da pilhaC;

proceder conforme linha 6 da tabela de decisão *até que* pop

senão : colocar ch na expressão posfixa

fim;

EXEMPLO: IN = (A-(B+C)/D*E); OUT: ABC+D/E*-;

+		
(/	*
-		
(

pilhaC

Vamos considerar que a expressão infixa não contenha erros.

Versão 1 do algoritmo de conversão

para cada caractere ch da expressão Infixa repita

caso ch seja

```

'(' : colocar ch na pilhaC;
'+', '-', ': {
    ok ← false;
    repita
        se (pilhaC vazia) então colocar ch na pilhaC; ok ← true;
        senão {
            chp ← topo da pilhaC;
            se (chp = '(')
                então colocar ch na pilhaC; ok ← true;
                senão colocar chp na expressão posfixa; pop
        }
    até que ok;
'*', '/': {
    ok ← false;
    repita
        se (pilhaC vazia) então colocar ch na pilhaC; ok ← true;
        senão {
            chp ← topo da pilhaC;
            caso chp seja:
                '(', '+', '-', ': colocar ch na pilhaC; ok ← true;
                '*', '/' : colocar chp na expressão posfixa; pop
            fim;
        }
    até que ok;
'} : enquanto (pilhaC não vazia) faça
    {chp ← topo da pilhaC;
    {pop; colocar chp na expressão posfixa;
')' : se (pilhaC não vazia)
    então {
        chp ← topo da pilhaC; ok ← false;
        repita
            se (chp = '(') então pop; ok ← true;
            senão colocar chp na expressão posfixa; pop
        até que ok;
    senão : colocar ch na expressão posfixa;
fim;

```

AValiação DE UMA EXPRESSÃO POSFIXA. Na avaliação de uma expressão posfixa, basta varrer a expressão a partir da posição inicial e efetuar cada operação encontrada com os valores que antecedem imediatamente o operador encontrado e substituir operandos e operador pelo resultado da operação.

Exemplo: ABC+D/E*-

Tabela de valores:

A	B	C	D	E
5	7	2	5	4

1) Primeira operação executada: $B + C = 7 + 2 = 9 = R$

ABC+D/E*-

2) Segunda operação executada: $R / D = 9 / 5 = 1.8 = S$

ARD/E*-

3) Terceira operação executada: $S \times E = 1.8 \times 4 = 7.2 = T$

ASE*-

4) Quarta operação executada: $A - T = 5 - 7.2 = -2.2$

AT-

Resultado final = -2.2

A idéia usada para fazer a avaliação é a seguinte: varrer a posfixa a partir da primeira posição, ao encontrar um identificador, colocar o valor correspondente (está na tabela de valores) na pilha de números (reais); ao encontrar um operador, retirar v1 e v2 da pilha (a operação é binária), efetuar a operação na ordem v2 op v1 e colocar o resultado da operação na pilha. Ao terminar a expressão posfixa, o resultado é o único valor contido na pilha.

Exemplo: ABC+D/E*-

Tabela de valores:

A	B	C	D	E
5	7	2	5	4

2	5	4		
7	9	1.8	7.2	
5				-2.2

pilhaV

Esboço do algoritmo de avaliação:

nP ← tamanho da expressão posfixa

para cada caractere ch da expressão posfixa, da posição 1 até nP-1, repita

se (ch é ident) então colocar valor de ch na pilhaV

senão v1 ← topo da pilhaV; pop; v2 ← topo da pilhaV; pop;

v ← operar v2 ch v1; colocar v na pilhaV;

valor da posfixa ← topo da pilhaV;

ATIVIDADE – POSFIXA

1) Escrever uma versão preliminar do algoritmo de conversão de infixa para posfixa, admitindo que a cadeia infixa possa conter erros posicionais.

2) Implementar o tipo ListaChar, de acordo com a declaração dada a seguir e com as operações usuais de lista.

```
#define Max 100;
typedef struct{
    int tamanho;
    char elemento[Max];
} ListaCHAR;
```

Implementar o tipo PilhaChar para processar uma pilha de caracteres, com as operações usuais do tipo pilha.

Implementar, para o tipo ListaChar, a função converterPosfixa que realiza a conversão de uma expressão infixa para a notação posfixa.

3) Implementar o tipo PilhaNumeros para processar uma pilha de números reais, com as operações usuais do tipo pilha.

Implementar o tipo Tabela para processar uma lista de identificadores e respectivos valores reais (identificadores = letras maiúsculas do alfabeto latino) com as funções usuais do tipo lista.

Implementar uma função avaliarPosfixa, para o tipo ListaChar, que realiza a avaliação de uma expressão posfixa.

4) Escrever as expressões na notação posfixa:

1	$A + B - C \times D;$	
2	$X - (Y - (A + B \times Z) + (D/H) \times X);$	
3	$A \times B - H;$	
4	$A - B - H;$	
5	$A + B / H;$	
6	$A \times B / H;$	
7	$X - (Y + A \times Z);$	
8	$X - (Y + A) \times Z;$	
9	$X - (Y - (A + B \times Z) + D / H) \times X;$	
10	$(X - Y - (A + B) \times (Z - D) / H \times X);$	
11	$X - Y + Z;$	
12	$X - (Y + Z);$	
13	$A \times B + C / D - H \times I;$	