## **Infix to Postfix Conversion**

To convert infix expression to postfix the operators' priority is as shown in the following table.

operator	Priority
^	3
*,/	2
+,-	1

Let the infix expression is ((a+b\*c-d ^ b/a)-e+b/e) +f; convert this infix expression into postfix using stack.

<u>Infix</u>	<u>Stack</u>	<u>Postfix</u>
((a+b*c-d ^ b/a)-e+b/e) +f		
$(a+b*c-d \wedge b/a)-e+b/e)+f$	(	
$a+b*c-d ^b/a)-e+b/e)+f$	((	
+b*c-d ^ b/a)-e+b/e) +f	((	a
b*c-d ^ b/a)-e+b/e) +f	((+	a
$*c-d \wedge b/a$ )- $e+b/e$ ) +f	((+	ab
$c-d ^b/a)-e+b/e)+f$	((+*	ab
$-d \wedge b/a$ ) $-e+b/e$ ) $+f$	((+*	abc
$d \wedge b/a$ )- $e+b/e$ ) +f	((-	abc*+
$^b/a)-e+b/e)+f$	((-	abc*+d
b/a)- $e+b/e$ ) +f	((-^	abc*+d
/a)- $e$ + $b/e$ ) + $f$	((-^	abc*+db
a)-e+b/e)+f	((-/	abc*+db ^
)-e+b/e)+f	((-/	abc*+db ^ a
-e+b/e)+f	(	abc*+db ^ a/-
e+b/e)+f	(-	abc*+db^ a/-
+b/e)+f	(-	abc*+db^ a/-e
b/e) +f	(+	abc*+db ^ a/-e-
/e) +f	(+	abc*+db ^ a/-e-b
e) +f	(+/	abc*+db^ a/-e-b
) +f	(+/	abc*+db^ a/-e-be

## **Postfix Evaluation**

Suppose a=2, b=4, c=3, d=1, e=2 and f=1, we can evaluate the postfix expression  $abc*+db ^ a/-e-be/+f+$  using the given values.

postfix expression	<u>stack</u>	<u>operation</u>
abc*+db ^ a/-e-be/+f+		
$bc*+db ^a-e-be/+f+$	a	
$c*+db ^a/-e-be/+f+$	ab	
*+db ^ a/-e-be/+f+	а <del>b</del> е	
$+db \wedge a/-e-be/+f+$	<del>a 12</del>	b*c=4*3=12
$db \wedge a/-e-be/+f+$	14	a+12=2+12=14
b ^ a/-e-be/+f+	14 d	
^ a/-e-be/+f+	14 <del>d -b</del>	
a/-e-be/+f+	14 1	d^ b=1 ^ 4=1
/-e-be/+f+	14 <del>1-a</del>	
-e-be/+f+	14 1/2	1/a=1/2=1/2
e-be/+f+	27/2	14- 1/2 = 27/2
-be/+f+	<del>27/2 e</del>	
be/+f+	23/2	27/2-2= 23/2
e/+f+	23/2 b	
/+ <b>f</b> +	23/2 <del>b e</del>	
+f+	<del>23/2 2</del>	b/e=4/2=2
f+	27/2	23/2+2=27/2

29/2

$$27/2+f=27/2+1=29/2$$

Result: 29/2

We can verify the result of postfix expression in the following way.

$$((a+b*c-d \wedge b/a)-e+b/e)+f$$

$$=((2+4*3-1^4/2)-2+4/2)+1$$

$$=((2+4*3-1/2)-2+4/2)+1$$

$$=((2+12-1/2)-2+4/2)+1$$

$$=((14-1/2)-2+4/2)+1$$

$$=(27/2-2+4/2)+1$$

$$=(27/2-2+2)+1$$

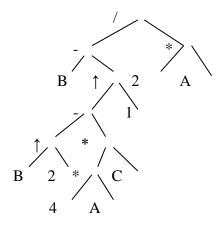
$$= 27/2 + 1$$

$$= 29/2$$
 ans

## **Binary Expression Tree**

A binary expression tree is a specific application of a binary tree to evaluate certain expressions. For example, consider the expression In-order (Infix notation)  $(B-(B\uparrow 2-4*A*C) \uparrow 1) / (2*A)$ 

As binary expression tree we can denote it as the following figure.



Post-order (postfix notation): BB2↑4A\*C\*-1↑-2A\*/

Pre-order (prefix notation) :  $/-B\uparrow-\uparrow B2**4AC1*2A$ 

Let us consider A=2, B=1, C=2.

Reverse the expression of Prefix: A2\*1CA4\*\*2B↑-↑B-/

Let us consider A=2, B=1, C=2.			
Reverse expression of Prefix	Stack	Operation	
A2*1CA4**2B↑-↑B-/			
2*1CA4**2B↑-↑B-/	A		
*1CA4**2B↑-↑B-/	A 2		
1CA4**2B↑-↑B-/	4	2*2=4	
CA4**2B↑-↑B-/	4 1		
A4**2B↑-↑B-/	4 1 C		
4**2B↑-↑B-/	4 1C A		
**2B↑-↑B-/	4 1 C A 4		
*2B↑-↑B-/	41C8	2*4=8	
2B↑-↑B-/	4 1 16	2*8=16	
B↑-↑B-/	4 1 16 2		
↑-↑B-/	4 1 16 2 B		
-↑B-/	4 1 16 1	1 ↑ 2 = 1	
↑B-/	4 1 -15	1-16=-15	
B-/	4 -15	-15↑1=-15	
-/	4 -15 B		
/	4 16	1-(-15)=16	
	8	16/4	