



Evaluating Postfix



Evaluating Postfix

A+B infix

AB+ postfix

- Each operator in a postfix expression refers to the previous two operands.
- Each time we read an operand, we push it on a stack.
- When we reach an operator, we pop the two operands from the top of the stack, apply the operator and push the result back on the stack.

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input

6

op1

op2

value

stack

6

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2
/	8	2	4	1,3,4

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2
/	8	2	4	1,3,4
+	3	4	7	1,7

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2
/	8	2	4	1,3,4
+	3	4	7	1,7
*	1	7	7	7

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2
/	8	2	4	1,3,4
+	3	4	7	1,7
*	1	7	7	7
2	1	7	7	7,2

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2
/	8	2	4	1,3,4
+	3	4	7	1,7
*	1	7	7	7
2	1	7	7	7,2
↑	7	2	49	49

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2
/	8	2	4	1,3,4
+	3	4	7	1,7
*	1	7	7	7
2	1	7	7	7,2
↑	7	2	49	49
3	7	2	49	49,3

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2
/	8	2	4	1,3,4
+	3	4	7	1,7
*	1	7	7	7
2	1	7	7	7,2
↑	7	2	49	49
3	7	2	49	49,3
+	49	3	52	52

Evaluating Postfix

Evaluate 6 2 3 + - 3 8 2 / + * 2 ↑ 3 +

Input	op1	op2	value	stack
6				6
2				6,2
3				6,2,3
+	2	3	5	6,5
-	6	5	1	1
3	6	5	1	1,3
8	6	5	1	1,3,8
2	6	5	1	1,3,8,2
/	8	2	4	1,3,4
+	3	4	7	1,7
*	1	7	7	7
2	1	7	7	7,2
↑	7	2	49	49
3	7	2	49	49,3
+	49	3	52	52

Evaluating Postfix

```
Stack s;  
while( not end of input ) {  
    e = get next element of input  
    if( e is an operand )  
        s.push( e );  
    else {  
        op2 = s.pop();  
        op1 = s.pop();  
        value = result of applying operator 'e' to op1 and op2;  
        s.push( value );  
    }  
}  
finalresult = s.pop();
```