Postfix Evaluation

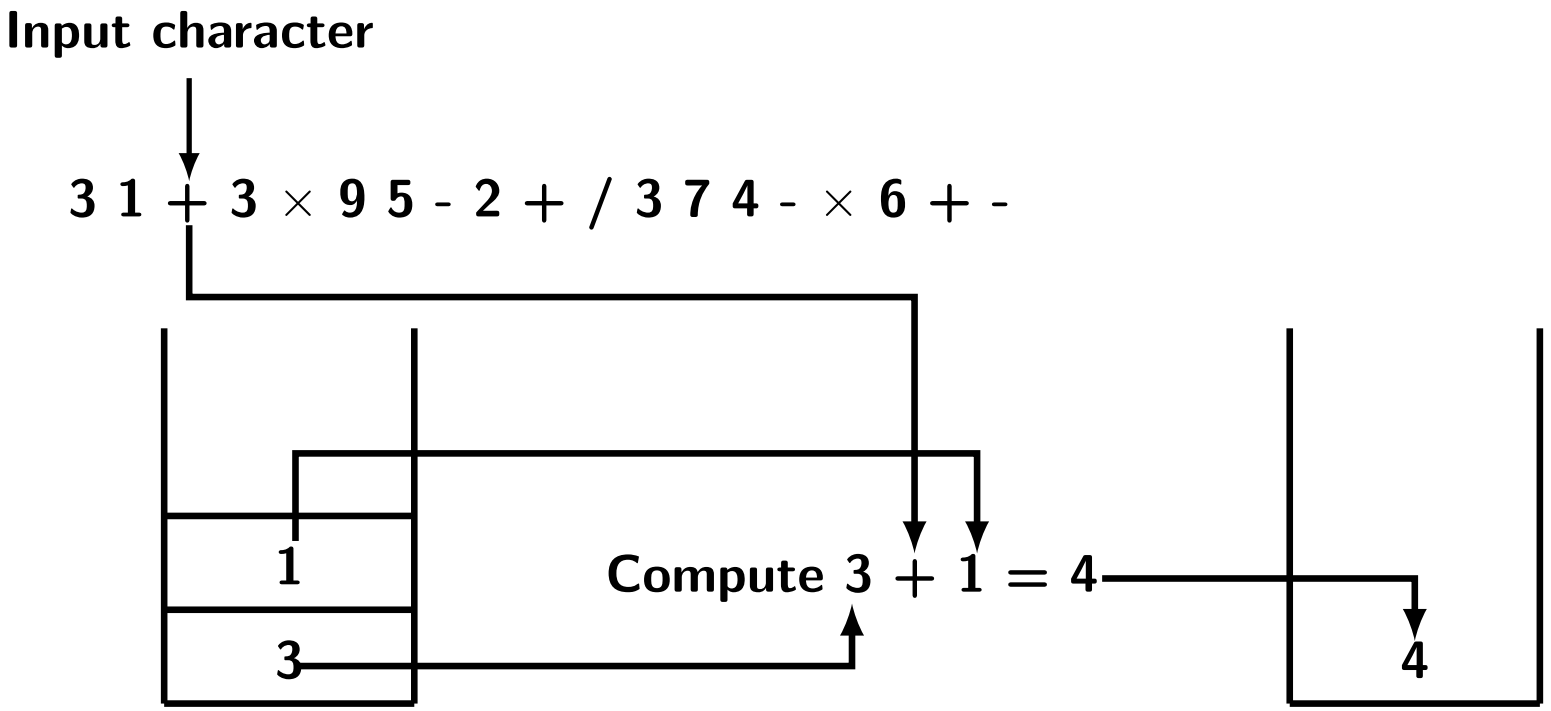
Postfix: 3 1 + 3 * 9 5 - 2 + / 3 7 4 - * 6 + -

- Observe that operator comes after operands.
- So, need to hold back operands until operator is encountered.
- It becomes easy with stack.
- A single left to right scan will do.
- For example: 3 and 1 are held back on the stack.
- When ”+” is encountered operation is applied on 3 and 1, the result is then pushed onto stack.

Postfix Evaluation

- Then next operand 3 is pushed.
- Now $\times$ is encountered, two operands on stack are 4 and 3 respectively.
- These are popped and multiplied, the result 12 is pushed to stack.
- Next 9, 5 are pushed, then 9-5 is computed and 4 is pushed.
- Note at this time 12 and 4 are on the stack, continue to get the result.

Postfix Evaluation



Infix to Postfix

- It is slightly complicated. You need to output numbers directly.
- Hold back the operator until two appropriate operands are on the output.
- To take care of precedence of evaluation, if a higher priority operator is found, lower priority operator should remain on the stack.
- Notice that in case of binary operators, an operator will be encountered after two operand are on output.

Infix to Postfix

- But, if the preceding operator's priority is low current operator must go to output.
- If the priorities are same then preceding operator (one already on stack) should go to the output.
  - Because the order of evaluation is from left to right for the operators having equal priorities.

Relationships Between Levels Precedence

- Levels of the internal nodes in the tree indicate their relative precedence in evaluation.
- Operations on top levels are evaluated after the operations in lower levels.
- The operation at the root is evaluated at the last.
- So, the evaluation actually starts bottom up or in the postorder manner.

Example

Infix Expression: (3 + 5) * (6 - 8/2)

Input	Stack	Output
1	(	
2	(	3
3	(+	3
4	(+	3 5
5	(	3 5 +
6		3 5 +
7	*	3 5 +
8	*(	3 5 +

Input	Stack	Output
9	*(	3 5 + 6
10	*(-	3 5 + 6
11	*(-	3 5 + 6 8
12	*(-/	3 5 + 6 8
13	*(-/	3 5 + 6 8 2
14	*(-	3 5 + 6 8 2 /
15	*(	3 5 + 6 8 2 / -
16	*	3 5 + 6 8 2 / -

Final result: 3 5 + 6 8 2 / - *

Evaluation of Infix Expression

- Assume that operands of are single characters: A, B, C, D.
- Unary operations are not permitted.
- Permitted operations are: +, -, *, /, and ˆ (exponentiation)

Two stacks are used:

1. Stack 1: storing operands,
2. Stack 2: storing operators and ’(’.

Evaluation of Infix Expression

Scan the input until the end is reached, get one character and perform only one of the following steps:

1. If character is an operand, push it onto operand stack and return to .
2. If character is an operator, perform ”Operate()” procedure.

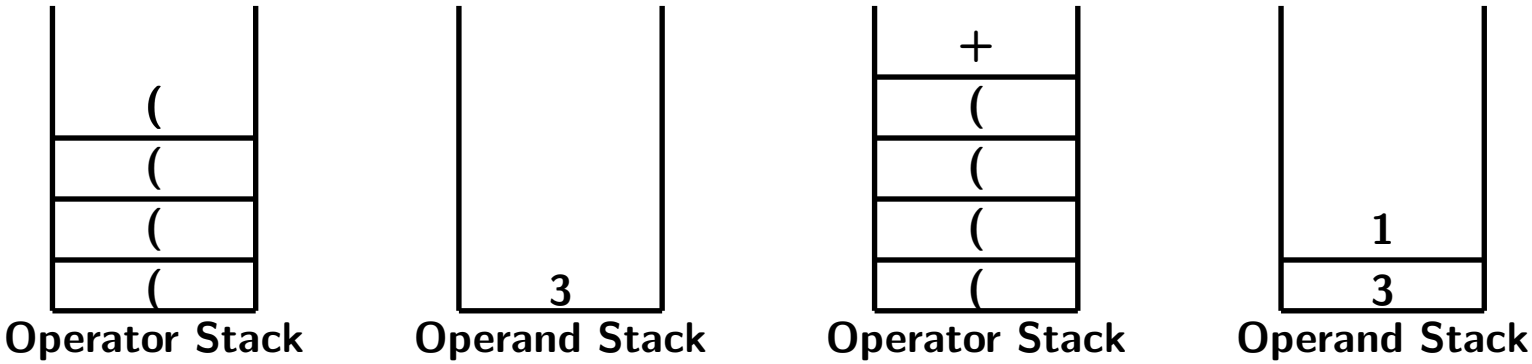
Procedure Operate()

It takes an operator as input and two stacks as arguments and performs as follows:

- If input operator is '(' push it onto operator stack and return.
- If input operator is ')' then repeat steps 1-4 below until corresponding '(' is reached. Then pop '(', discard it, and return.
- Else while input operator's precedence is less than or equal to precedence of the operator on the top of operator stack
  1. Pop one operand from operand stack call it "value2".
  2. Pop operator from top of the operator stack.
  3. Pop another operand from operand stack call it "value1".
  4. Compute value1 op value2 and push the result onto operand stack.

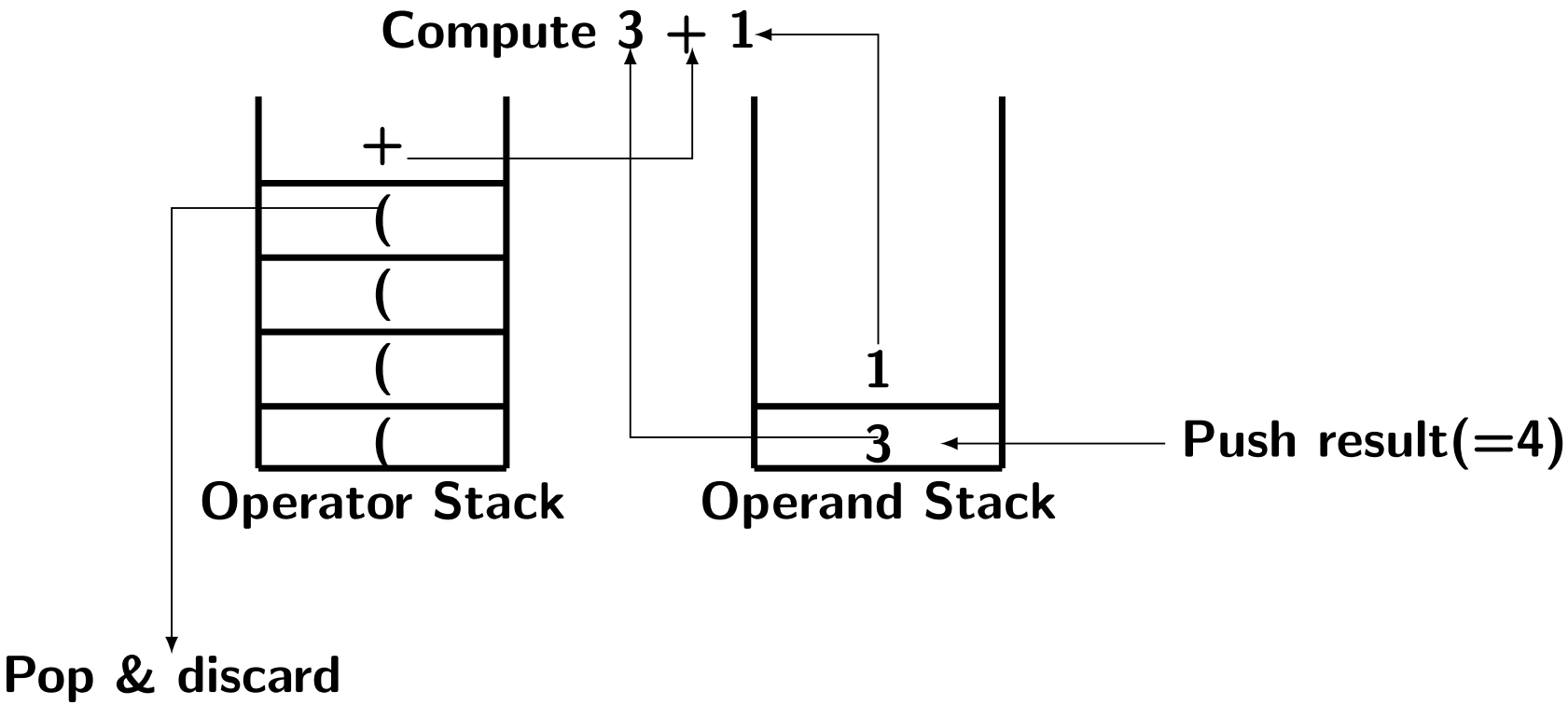
  Push the input operator in the operator stack and return.

Example



- Next operator is ')', so procedure Operate will executed.
- '+' popped from operator stack, 3+1 is computed and pushed back

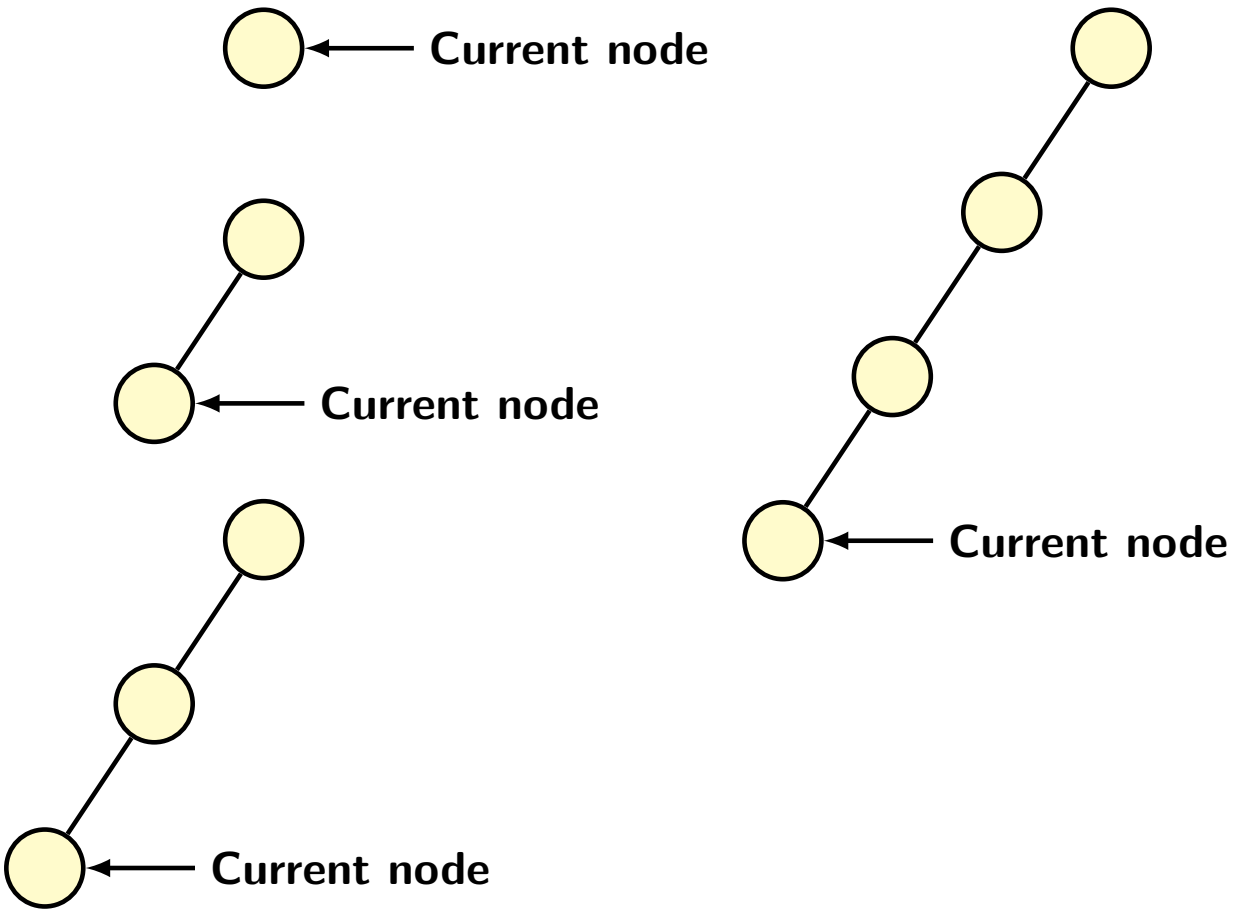
Example


Compute 3 + 1
+
(
(
(
(
1
3
Push result(=4)
Operator Stack
Operand Stack
Pop & discard

Building an Expression Tree

1. If the current char (token) is '(', add a new node as the left child of the current node and descend down to the left child.
2. If the current token is in: $<$ '+', '-', '/', '*'$>$ then set the value of current node to the corresponding operator. Add a new node as right child and descend down to right child.
3. If the current token is a number n, set the value of the current node to n and go to parent of the current node.
4. If the current char is ')', then go to parent of the current node.

Result of Repeated Applications of Rule 1



Infix: (((((3+1)× 3))/((9-5)+2))+((3×(7-4)) + 6))

Application of Rule 3