Application of Stack

1. Infix to Postfix Conversion

Algorithm

1.Scan the input string (infix notation) from left to right. One pass is sufficient.

2. If the next symbol scanned is an operand, it may be immediately appended to the postfix string.

3. If the next symbol is an operator,

i. Pop and append to the postfix string every operator on the stack that

a. is above the most recently scanned left parenthesis, and

b. has precedence higher than or is a right-associative operator of equal precedence to that of the new operator symbol.

ii. Push the new operator onto the stack.

4. When a left parenthesis is seen, it must be pushed onto the stack.

5. When a right parenthesis is seen, all operators down to the most recently scanned left parenthesis must be popped and appended to the postfix string. Furthermore, this pair of parentheses must be discarded.

6. When the infix string is completely scanned, the stack may still contain some operators. All the remaining operators should be popped and appended to the postfix string.

Question

Infix Expression= A+(B*C-(D/E-F)*G)*H

Find out the Corresponding postfix Expression.

Solution

Stack	Input	Output
Empty	A+(B*C-(D/E-F)*G)*H	-

Empty	+(B*C-(D/E-F)*G)*H	A
+	(B*C-(D/E-F)*G)*H	A
+(	B*C-(D/E-F)*G)*H	A
+(	*C-(D/E-F)*G)*H	AB
+(*	C-(D/E-F)*G)*H	AB
+(*	-(D/E-F)*G)*H	ABC
+(-	(D/E-F)*G)*H	ABC*
+(-(	D/E-F)*G)*H	ABC*
+(-(	/E-F)*G)*H	ABC*D
+(-(/	E-F)*G)*H	ABC*D
+(-(/	-F)*G)*H	ABC*DE
+(-(-	F)*G)*H	ABC*DE/
+(-(-	F)*G)*H	ABC*DE/
+(-(-	)*G)*H	ABC*DE/F
+(-	*G)*H	ABC*DE/F-
+(-*	G)*H	ABC*DE/F-
+(-*	)*H	ABC*DE/F-G
+	*H	ABC*DE/F-G*-

+*	H	ABC*DE/F-G*-
+*	End	ABC*DE/F-G*-H
Empty	End	ABC*DE/F-G*-H*+