Applications of Stack Contd…

Prof. Harish D.G.
Dept. of Computer and IT
College of Engineering,Pune
www.harishgadade.com

Infix to prefix Conversion

Steps:

1. Reverse the infix expression.
2. Make every ‘(‘ as ‘)’ and every ‘)’ as ‘(‘
3. Convert modified expression to postfix form. [Use algorithm given for postfix conversion]
4. Reverse the postfix expression.

Infix Expression : a+(b-c)
Reverded Infix Expr :) c - b) + a

Infix to prefix Conversion

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Input String : (c - b) + a #

Infix to prefix Conversion

Input String : (c - b) + a #

CH	Stack	Output String
(	(	

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Infix to prefix Conversion

Input String : (c - b) + a #

CH	Stack	Output String
(	(	
c	(	c

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Infix to prefix Conversion

Input String : (c - b) + a #

CH	Stack	Output String
(	(	
c	(	c
-	(-	c

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Infix to prefix Conversion

Input String : (c - b) + a #

CH	Stack	Output String
(	(	
c	(	c
-	(-	c
b	(-	cb

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Infix to prefix Conversion

Input String : (c - b) + a #

CH	Stack	Output String
(	(	
c	(	c
-	(-	c
b	(-	cb
)		cb-

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Infix to prefix Conversion

Input String : (c - b) + a #

CH	Stack	Output String
(	(	
c	(	c
-	(-	c
b	(-	cb
)		cb-
+	+	cb-

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Infix to prefix Conversion

Input String : (c - b) + a #

CH	Stack	Output String
(	(	
c	(	c
-	(-	c
b	(-	cb
)		cb-
+	+	cb-
a	+	cb-a

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Infix to prefix Conversion

Input String : (c - b) + a #

CH	Stack	Output String
(	(	
c	(	c
-	(-	c
b	(-	cb
)		cb-
+	+	cb-
a	+	cb-a
#		cb-a+

Infix Expression : a+(b-c)

Reverded Infix Expr : (c - b) + a #

Postfix Expr : cb - a +

Reverded Postfix Expr : +a-bc

Postfix Expression : +a-bc

Examples

1. A + B
2. A + B – C
3. (A + B) * C
4. (A + B) * (C – D)
5. A * B + (C – D / E)
6. A – B / (C * D ^ E)
7. ((A + B) * C - (D – E)) ^ (F + G)
8. A ^ B * C – D + E / F / (G + H)
9. (((A / (B ↑ C)) + (D * E)) - (A * C))

Postfix to infix conversion

Requirements :

1. postfix expression
2. Stack to store operands and infix expression

Algorithm :

```
1.  CH = next input character from postfix string
2.  If CH = = operand
        PUSH CH in Stack
    else if CH = = operator
        op1 = POP
        op2 = POP
        infixstr = ( op2 CH op1 )
        PUSH infixstr in stack.
3.  If postfix expression is not over then go to step 1
4.  POP from stack and display infix expression
5.  Stop.
```

Examples

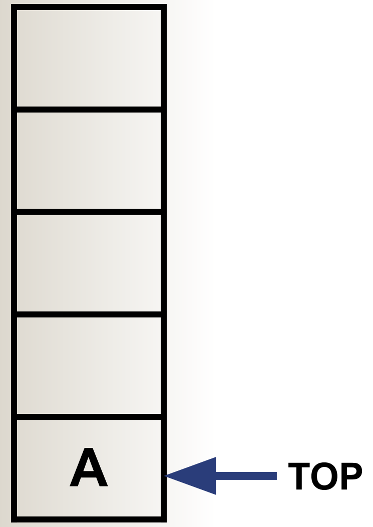
1. A B + C –
2. A B + C D E / - +
3. A B C D E ^ * / -
4. A B ^ C * D – E F / G H + / +
5. a b + c d - * a b c / * -

Examples:

A B + C -

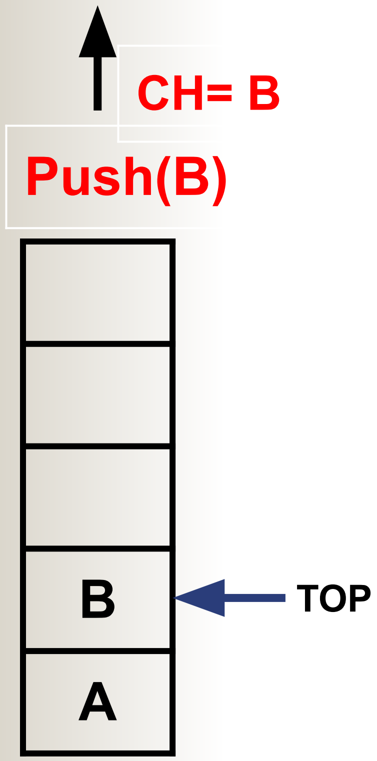
CH= A

Push(A)



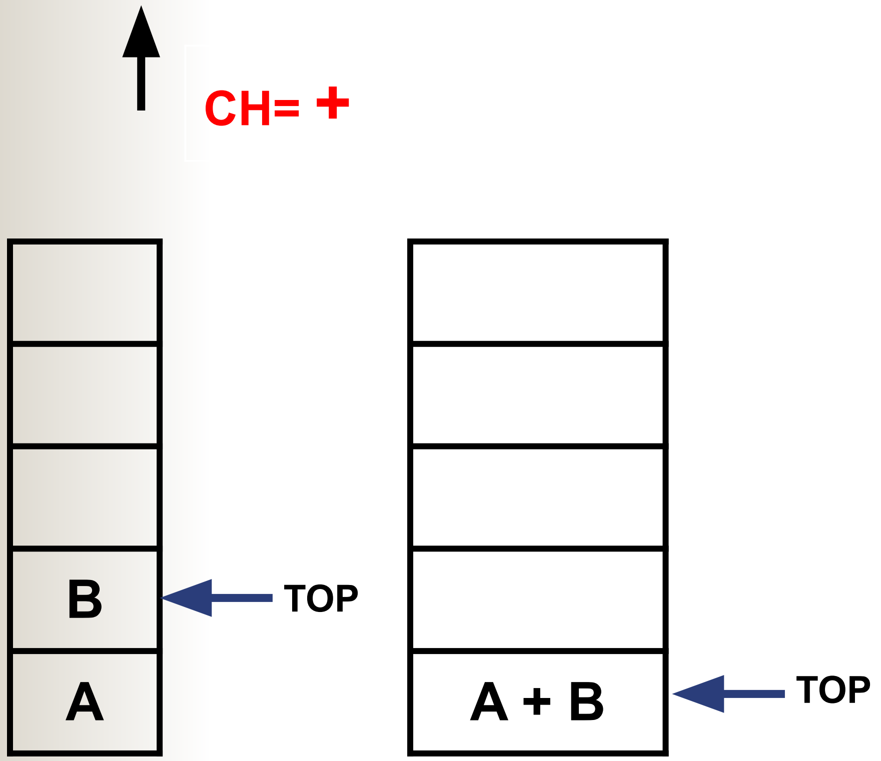
Examples:

A B + C -



Examples:

A B + C -



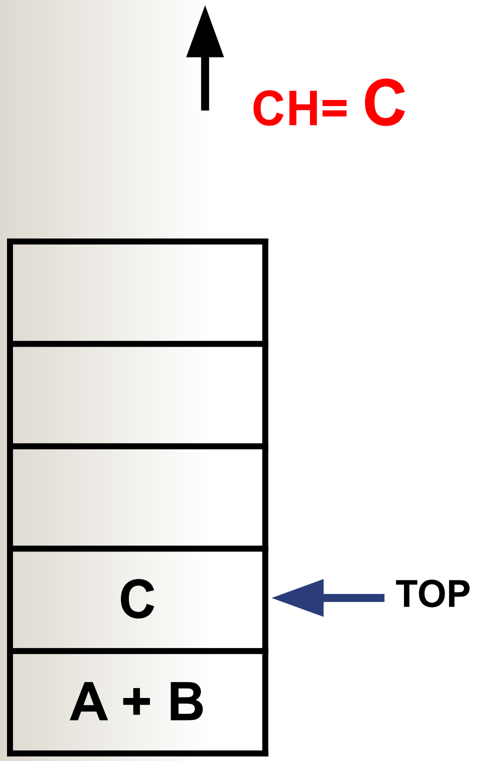
pop() top two elements i.e.
opernd_1 = B
opernd_2 = A

result = opernd_2 + opernd_1
result = A + B

push(A+B) back to stack

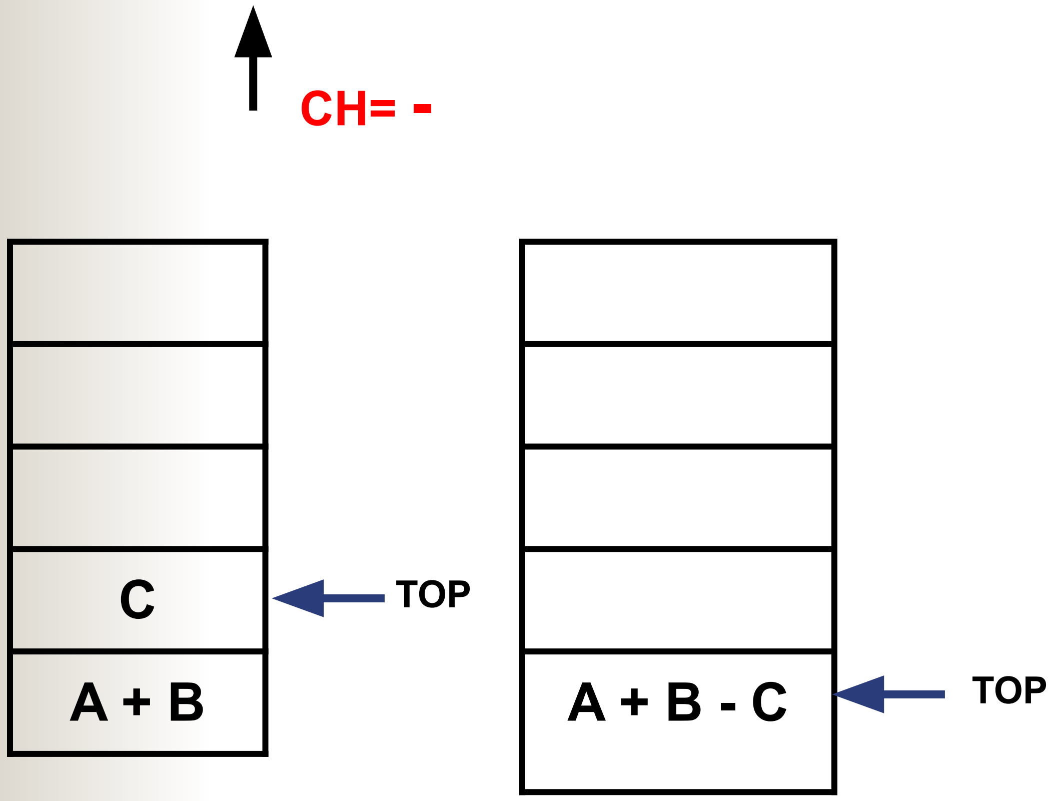
Examples:

A B + C -



Examples:

A B + C -



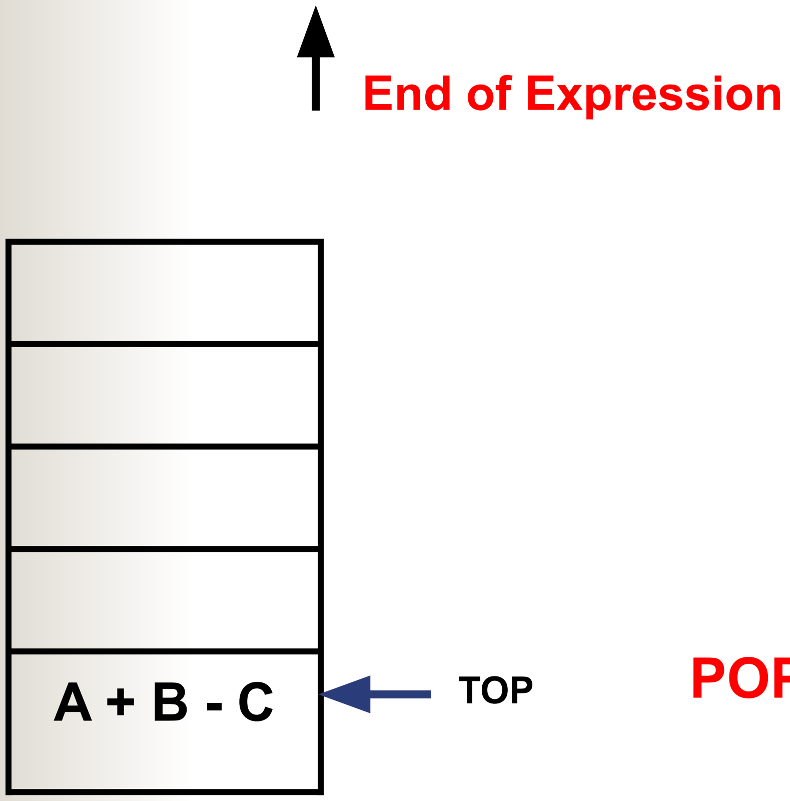
pop() top two elements i.e.
opernd_1 = C
opernd_2 = A + B

result = opernd_2 - opernd_1
result = A + B - C

push(A+B-C) back to stack

Examples:

A B + C -



POP ()

Result = A+B-C

Postfix to prefix Conversion

Requirements :

1. Postfix string
2. stack to store operands
3. Buffer to store prefix expression

Algorithm:

```
1. CH = next input character from postfix string
2. If CH = operand then
            PUSH it in stack
   else
            op1 = POP
            op2 = POP
            prefixstr = CH op2 op1
            PUSH prefix string into stack.
3. If postfix string not over then goto step 1.
4. POP prefix string from stack and display.
5. Stop
```

e.g. consider a b c * +

convert it into prefix form

abc*+

a(b*c)+

(a+(b*c))

Examples:

1. A B + C –

2. A B + C D - *

3. A B * C D E / - +

Prefix to infix conversion

Steps:

1. Prefix expression is scanned from right to left.
2. When CH = operand, then PUSH it on to the stack.
3. If CH=operator, then POP top two elements from stack , merge these two operands and operator to create a infix expression.
4. PUSH this expression back on to the stack.

e. g. Consider the prefix expression

+ a * b c

Convert it into infix form.

Examples :

1. - + A B C

2. * + A B – C D

3. + * A B – C / D E

4. - A / B * C ^ D E

5. ^ - * + A B C – D E + F G

Prefix to Postfix conversion

Steps:

1. Prefix expression is scanned from right to left.
2. When CH = operand, then PUSH it on to the stack.
3. If CH = operator, then POP top two elements from stack , merge these two operands and operator to create a infix expression.
4. PUSH this expression back on to the stack.