CS & IT ENGINEERING

Data Structure

Stack and Queues

Chapter- 4

Lec- 03

By- Pankaj Sharma sir

Stack-III

infix to postfix using stack

infix : 2 + 3

Postfix : 2 3 +

Prefix : + 2 3

infix : 2 + 3 × 4

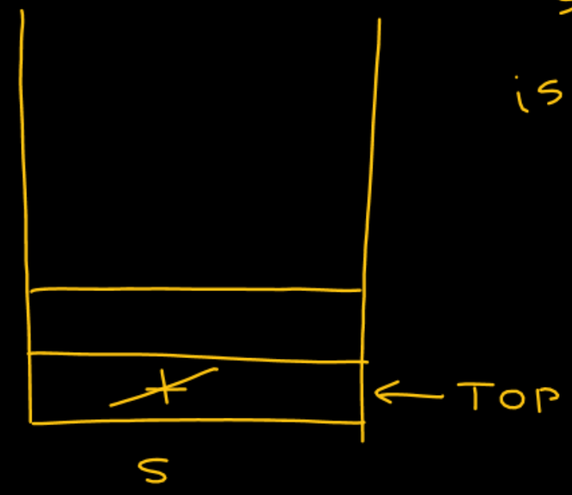
Postfix : 2 3 4 × +

Prefix : + 2 × 3 4

Ex1:

infix : 2 + 3 end

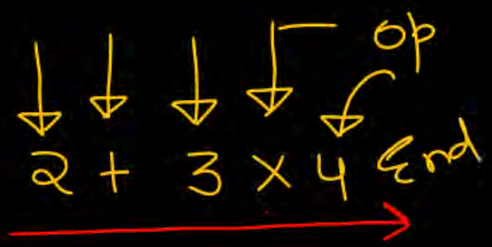
O/P : 23+



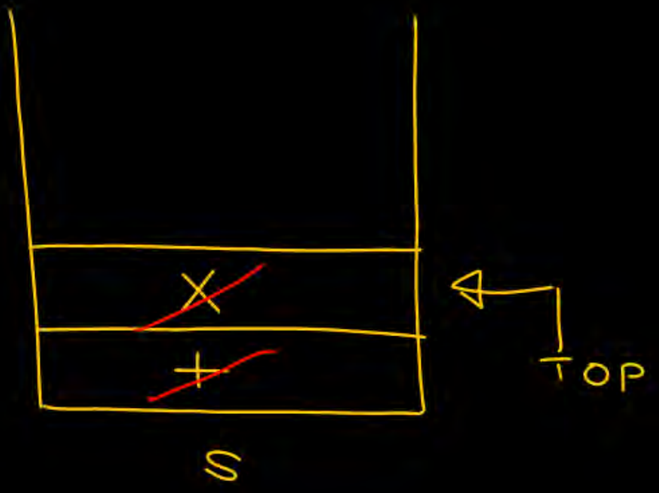
scan element is operator

$\Longrightarrow$ and stack is empty

Ex2:



Postfix: 2 3 4 x +



Stack-TOP Scan

+ < x

Ex3:

o/p : a b + c -

+ -

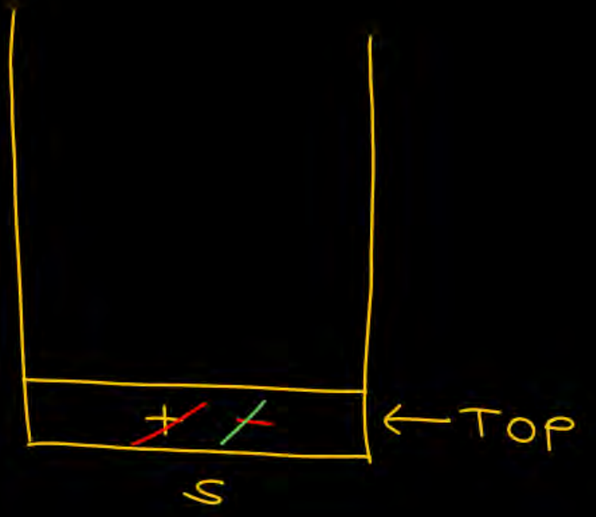
Postfix : a b + c - ✓

⇒ -

a + b - c

[a b +] - c

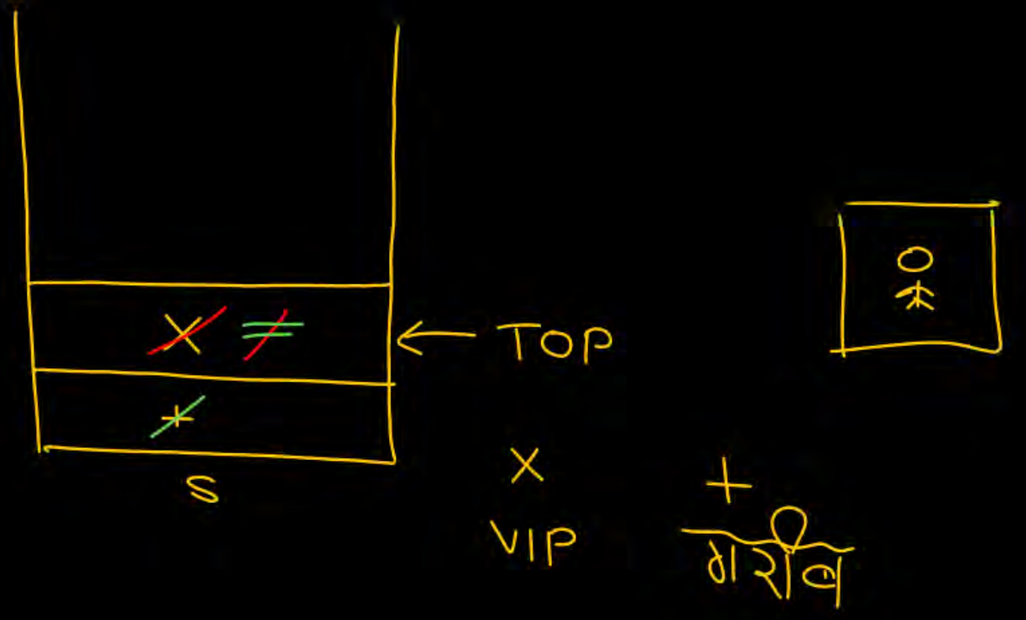
a b + c - ✓



Ex4

i/p : a + b x c / d (End)

o/p : a b c x d / +



Ex:S →

i/p : 2 + (3 × 4 − 6/2)

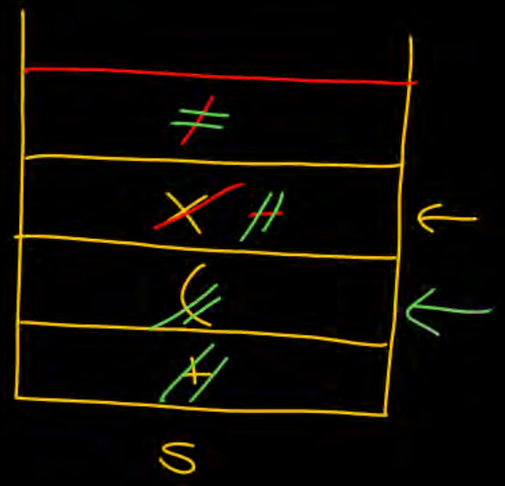
Right Bar

← End

o/p : 2 3 4 × 6 2 / − +

left to right

left (⇒ Push



Ex6

i/p : a + b x c / d - e ^ f ^ g + h End

o/p : a b c x d / + e f g ^ ^ - h +

^	R to L
x /	L to R
+ -	L to R

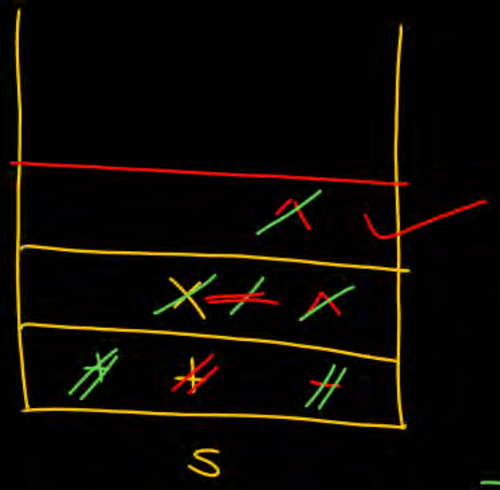
a + b x c / d - e ^ f ^ g + h

a + b x c / d - e ^ [f g ^] + h

a + b x c / d - [e f g ^ ^] + h

a + [b c x] / d - [e f g ^ ^] + h

a + [b c x d /] - [e f g ^ ^] + h



[a b c x d / +] - [e f g ^ ^] + h

[a b c x d / + e f g ^ ^ -] + h

a b c x d / + e f g ^ ^ - h +

infix to prefix

- without stack
- using stack

infix : a + b x c

a + [x b c]

+ a x b c

[+ a x b c] - [/ d ^ e ^ f g]

- + a x b c / d ^ e ^ f g

a + b x c - d / e ^ f ^ g

a + b x c - d / e ^ [^ f g]

a + b x c - d / [^ e ^ f g]

a + [x b c] - d / [^ e ^ f g]

a + [x b c] - [/ d ^ e ^ f g]

$[+ab] - d/(c-d+e)$

$[+ab] - d/[+-cde]$

$[+ab] - [/d+-cde]$

$-+ab/d+-cde$

$(a+b) - d/((c-d+e))$

$[-cd]+e$

$(a+b) - d/[+-cde]$

$(a+b) - [/d+-cde]$

$[+ab] - [/d+-cde]$

$-+ab/d+-cde$

$$(a + b \times c) - d / (e \uparrow f \uparrow g + h)$$

$a + [\times bc]$

$[+a \times bc] - d / [+ \uparrow e \uparrow f g h]$

$e \uparrow [\uparrow f g] + h$

$[\uparrow e \uparrow f g] + h$

$[+a \times bc] - [/d + \uparrow e \uparrow f g h]$

$- + a \times bc / d + \uparrow e \uparrow f g h$

infix to prefix using stack

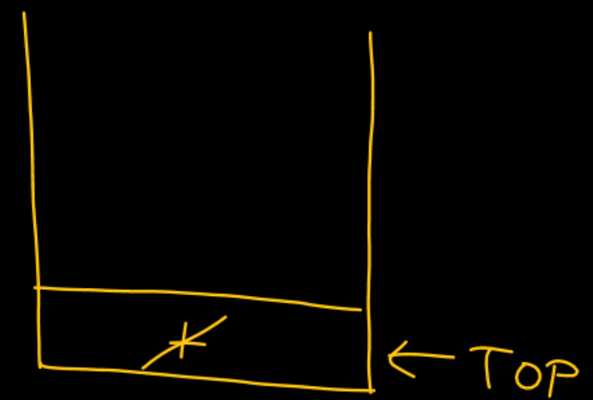
Ex 1

infix : 2 + 3

Reverse of infix : 3 + 2 End

O/P : 3 2 +

Reverse : + 2 3



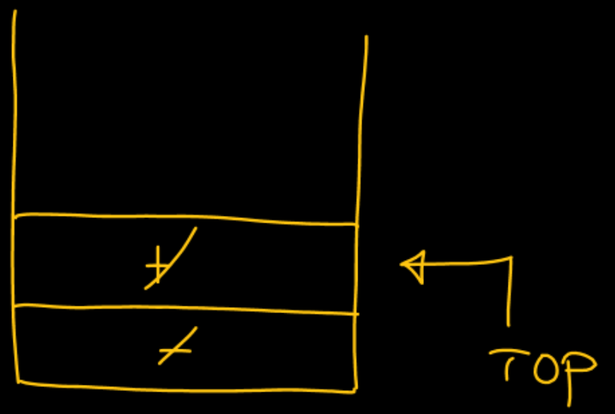
Ex2:

i/p : a + b - c

Reverse of infix : c - b + a End

o/p : c b a + -

Reverse o/p : - + a b c



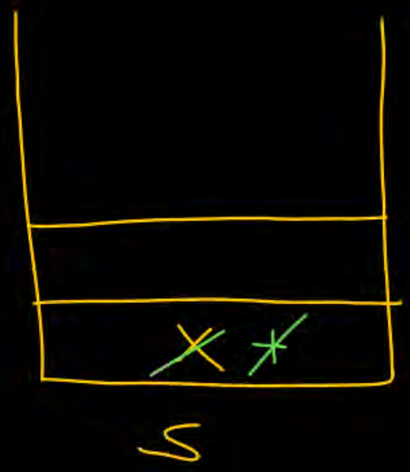
Ex3:

infix : 2 + 3 X 4

Reverse of infix : 4 X 3 + 2 end

O/P : 4 3 X 2 +

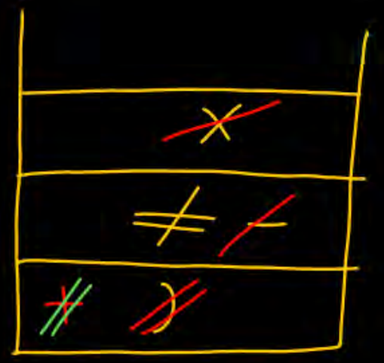
Reverse O/P : + 2 X 3 4



Ex4:

infix: a + (b x c - d / e)

Reverse of infix:) e / d - c x b (+ a End



O/P: e d / c b x - a +

Reverse of O/P: + a - x bc / de

a + (b x c - d / e)

[x bc] - d / e

[x bc] - [/ de]

a + [- x bc / de]

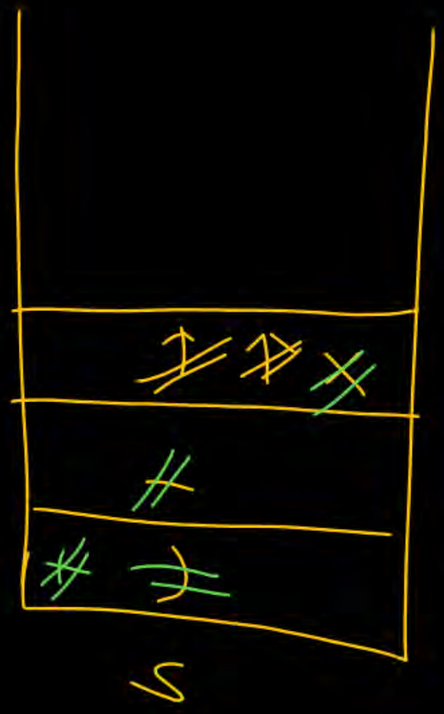
+ a - x bc / de

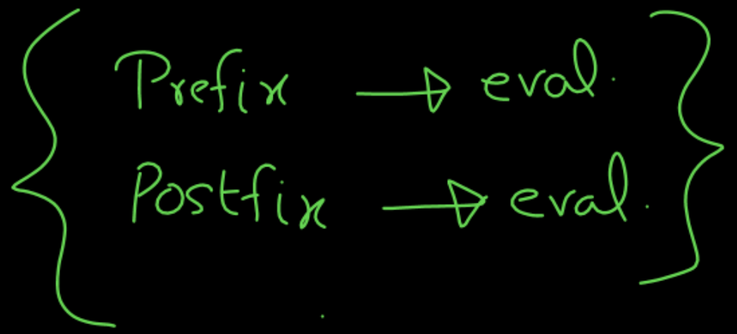
Ex5 infix : A + (B X C ↑ D ↑ E - F)

Reverse infix :) F - E ↑ D ↑ C X B (+ A

O/P : F E D ↑ C ↑ B X - A +

Reverse O/P : + A - X B ↑ C ↑ D E F




Prefix → eval.
Postfix → eval.


THANK YOU GW
SOLDIERS !