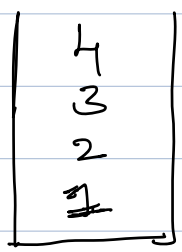
Stacks intro

monotonic stack

LIFO ⇒ last in first out.

1) Stack of plates.
2) " " chairs
3) " " clothes
4) Bangles



↓ 1, 2, 3, 4

→ 4 3 2 1

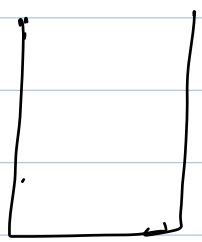
Operations supported by Stacks

1) push(x) : pushes x to the top of stack
2) pop() : removes topmost element and returns it.
3) top() / peek() : returns topmost element
4) size() : size of stack.
5) isEmpty() → returns true if empty.

Application of Stack

1) Recursion

a b a



2) Undo / Redo

3) Browsers navigation.

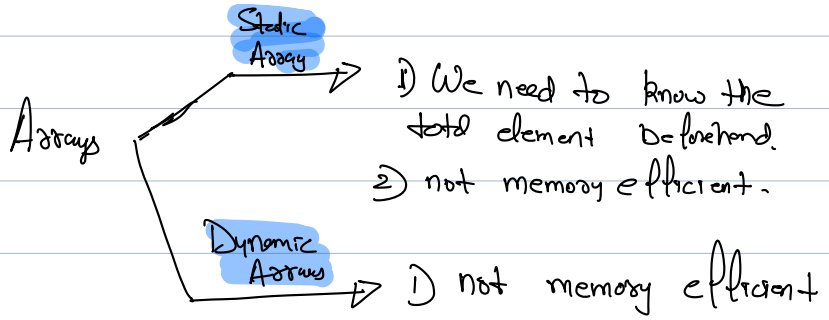
4) Bracket check

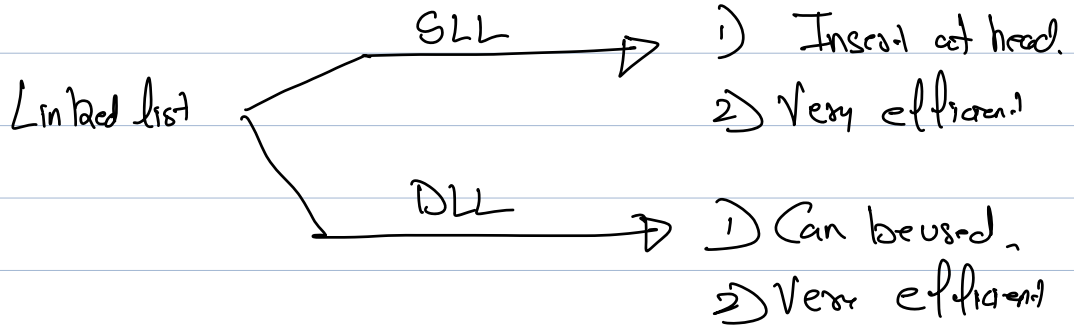
[{ ( ) ( ) }]

5) Expression Evaluation.

(4+3)*2 + 5/2

Implementation of Stack





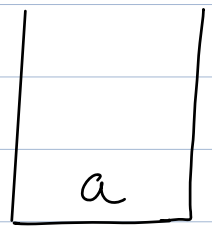
Q1 Given a string. Remove every consecutive duplicate pair of characters until there are no consecutive duplicate pairs.

Ex1 S: a c b b c k
⇒ a c c k
⇒ a k.

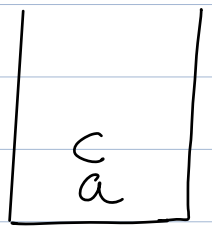
Ex2 S: a a a k
⇒ a k

Ex3 S: a b c k k c b a d m m c ⇒ d c

S: a c b b c K

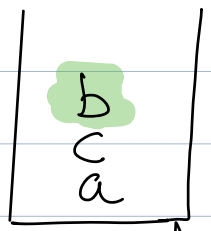


a c b b c K



a c b b c K

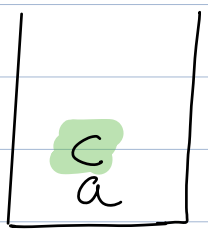
a c b b c K



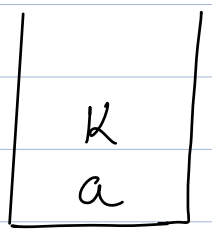
currentCle == topOfStack

→ pop & move forward without pushing

a c b b c K



a c b b c K



remove all element & store to ans

ans ⇒ "Ka"

reverse

ans = "aK"

TC: O(n)

SC: O(n)

Evaluating Expression :

Operators: +, −, *, /
Operands: the numerical number.

Infix: Operators b/w operands.

A+B

Postfix : operators come after operands

AB+

Infix —Stack→ postfix —Stack→ evaluated

Convert Infix to Postfix

1) A x B + C

ABX + C

ABxC+

2) A + B x C

A + BCx

ABCx+

3) 10 + 3x4 - 7

10 + 34x - 7

10 34x + - 7

1034x+7-

( )

/ x

+ -

→

4) 10/(4-2) x 6 + 9

10 / 42- x 6 + 9

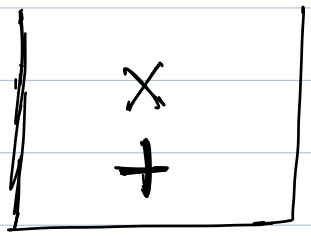
10 42-/ x 6 + 9

10 42-/6x + 9

10 42-/6x9+

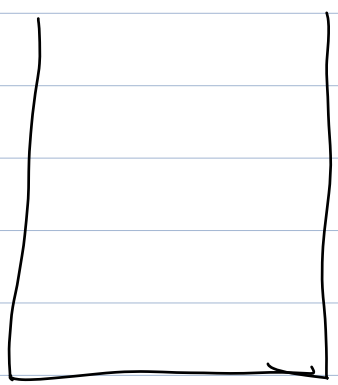
→ Relative order of operand remain the same.

Ex1 Infix : A + B x C.



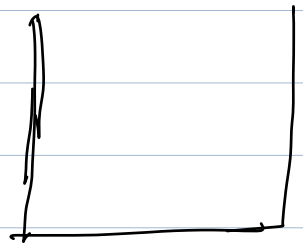
ans = "ABCx+"

Ex2 Infix : A x B + C.



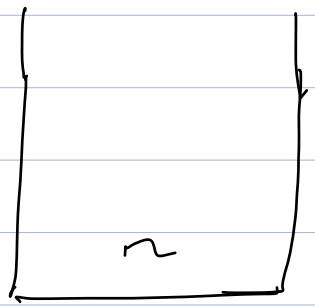
ans = "ABxC+

Ex3 Infix : 10 + 3 x 4 − 7.



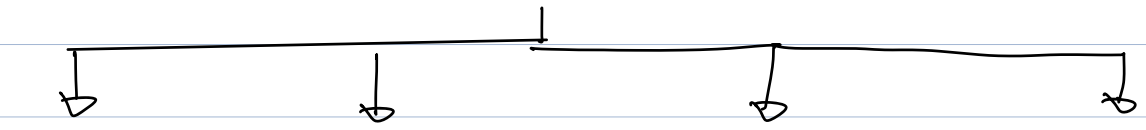
ans = " 10 3 4 X + 7 −

Ex4 10 x (3 + 4) + 5.



ans = 10 3 4 + x 5 +

Traverse the expression.



Operand.
↓
push to ans;

Operator.
↓
Pop till top of stack has lower priority operator.
Push the current operator.

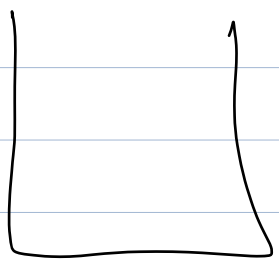
(
↓
push to stack

)
↓
Pop till you get an opening bracket

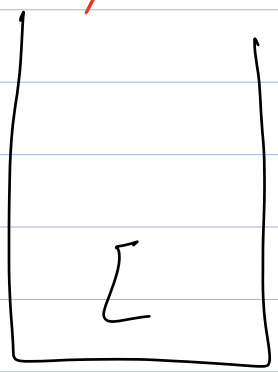
TC: O(n)
SC: O(n)

Q Bracket Check /

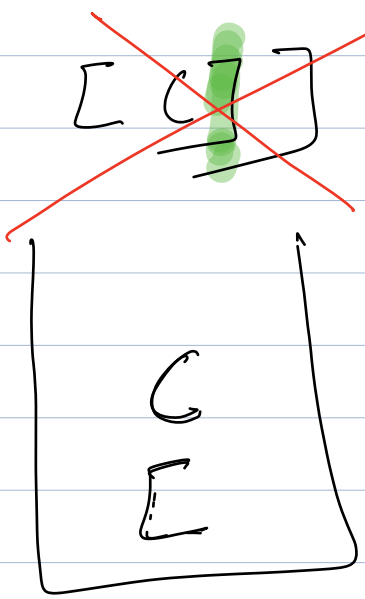
Ex1 [( )]



Ex [( )



Ex3 [( ])



=> False

$10 \times (3+4) + 5$.

Postfix $\Rightarrow$ 10 3 4 + × 5 +

(−)

· AB−

A − B