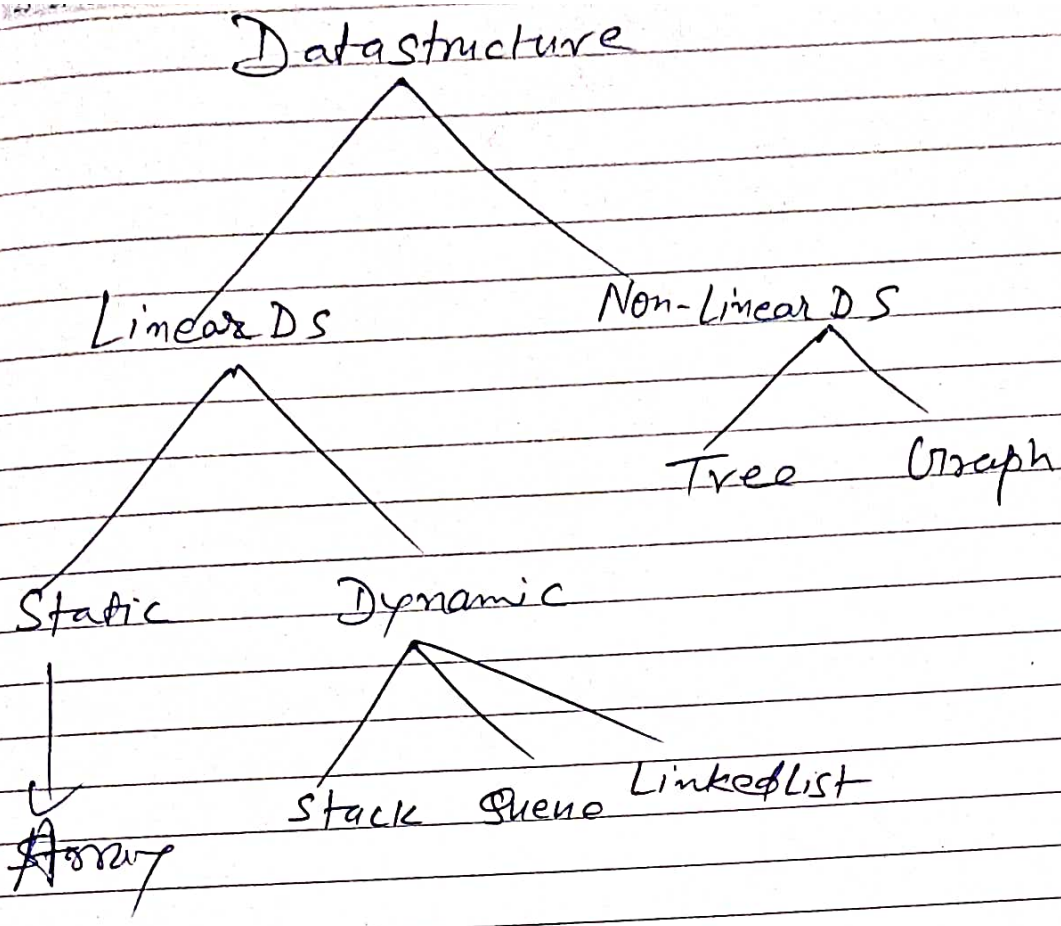


Datastructure :- Datastructure is a particular way of organized data in particular ~~format~~ in a computer that provides efficient way to access data and modification.

Linear DS :- ~~A linear datastructure have data~~ The Datastructure whose elements are arranged in sequential manner and each element is connected its previous and next elements. It is also called Single-Level DS.

NOTES : APPOINTMENTS : PHONE/E-MAIL

Non-Linear DS :- ~~Non-linear DS~~ The Datastructure whose data elements are not arranged sequentially or linearly. It is also called multilevel DS.

NOTES

DATE

~~STACK~~

~~A stack~~

Classification of Data structures

1. Simple Data structure

   a. Homogeneous DS

   A DS is said to be homogeneous, if the elements are ~~of~~ similar type.

   of

   एक ही

   => Array

   b. Non-Homogeneous DS

   A DS is said to be non-homogeneous, if the elements may not be the ~~of~~ similar type.

   of एक ही

   => Structure.

DATE

Q What are primitive and non-primitive datatypes?

A Primitive datatypes are those datatypes, which are not composed of other datatypes.

Non-primitive datatypes are those datatypes, which are composed of primitive datatypes.

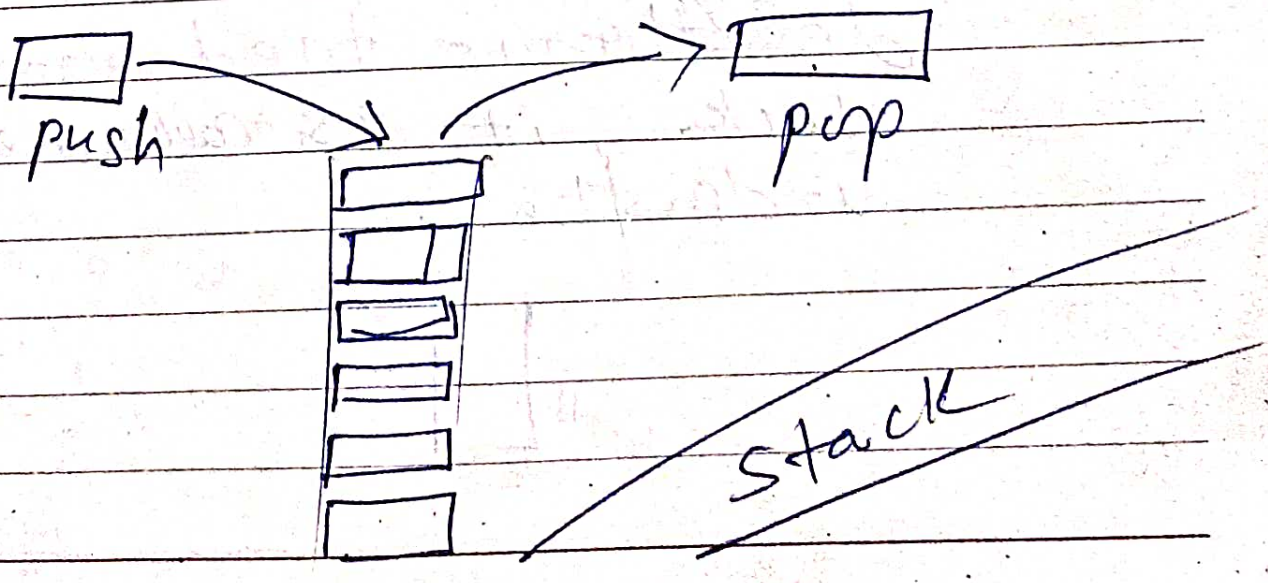
NOTES

DATE

Stack : A stack is a linear data structure implemented in LIFO (Last In First Out) manner. Where insertions and deletions are restricted to occur only at one end - Stack's top.

LIFO means element last inserted would be the first one to be deleted.

The Stack is also a dynamic data structure as it can grow (with increase in number of elements) or shrink (with decrease in number of elements).



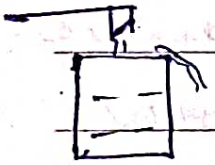
NOTES

DATE

Pushing :- An insertion in a stack is called pushing.

Popping - A deletion from a stack is called popping.

Overflow :- When a stack implemented as an array is full and no new element can be accomodated, it is called an overflow.



Underflow - When a stack is empty and an element's deletion is tried from the stack, it is called an underflow.

well

NOTES

Application of Stacks

DATE

polish string

polish string refers to infix prefix and postfix notation.

~~It is used for complex arithmetic expression into machine readable form using stack.~~ conversion of complex

It is used for convertion of complex arithmetic expression in high level language into machine readable form using stack.

Infix Notation: In this notation, the operator symbol is placed in between the operands.

e.g a+b

NOTES

DATE

prefix Notation

In this notation, the operator symbol is placed before its operands.

eg +ab

postfix Notation

In this notation, the operator symbol is placed after its operands.

eg ab+

* Evaluation order

I. Brackets or parenthesis

II Exponentiation or Exponent (↑ or ^)

III Multiplication or Division

IV Addition or Substraction

The operators with same priority are evaluated from left to right.

DATE

i.e [ ] ( ) ↑ * / + −

Arithmetic operator or ↑ * / + −

Logical operator ! && ||

NOT AND OR

Algorithm for push operation in stack

Insertion in a stack as an Array (pushing)

: Take an array stack[] with N element and item is the value to be push on the top of the stack. The variable Top is pointing to the topmost element of the stack

step

1. if top == N-1 then
2. print "Overflow!!"; exit from program
3. Else
4. top = top + 1
5. stack[top] = item
6. End

NOTES

DATE

e.g. stack[ ]

N = 4

I. item = 70

Top = 3

3	60
2	50
1	40
0	30

II. item = 70

Top = 2

3	
2	50
1	40
0	30

III item = 70

Top = -1

NOTES

2010

DATE

Algorithm for pop operation in stack

Take an array stack[] with N element and item is the varible to be store the deleted value from stack. The variable Top is pointing to the topmost element of stack.

step (I) if top == -1 (or top < 0) then

(II) print "Underflow !!" exit from program

(III) Else

(IV) item = stack[top]
(print or stack[top])

(V) ~~top~~ top = top - 1

(VI) End

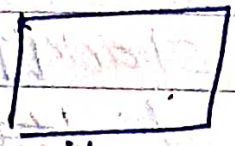
NOTES

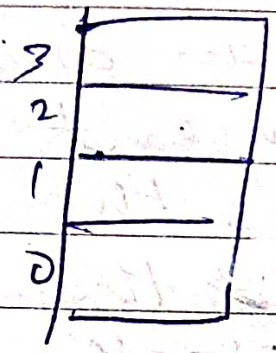
DATE

e-g stack[ ]

N=4

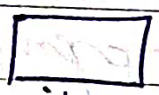
I

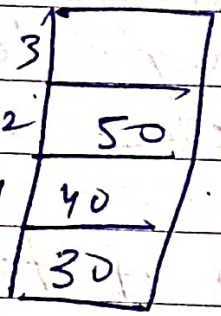




Top = -1

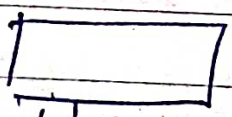
II





Top = 2

III



Top = 0

NOTES

Conversion

DATE

1) Expression Method

infix → prefix
infix → postfix

Ex (I) a+b*c

prefix	postfix
a+b*c	a+b*c
a+(*bc)	a+(bc*)
=+a*bc	abc*+

(II) a+b*c/d 2009

prefix	postfix
a+b*c/d	a+b*c/d
a+(*bc)/d	a+(bc*)/d
a+/*bcd	a+bc*d/
+a/*bcd	abc*d/+

(III) (a+b)/(c*d)

prefix	postfix
(+ab)/(*cd)	(ab+)/(cd*)
/+ab*cd	ab+cd*/

NOTES

(iv) $(A+(B*C))/(C-(D*B))$ DATE 2010

postfix

$(A+(BC*))/(C-(DB*))$

$(ABC*+)/(CDB*-)$

$ABC*+CDB*-/$

prefix

$(+A*BC)/(-C*DB)$

$/+A*BC-C*DB$

(v) $(A+B)*C/D\uparrow F$

prefix

$(+AB)*C/(\uparrow DF)$

$(*+ABC)/(\uparrow DF)$

$/*+ABC\uparrow DF$

postfix

$(AB+)*C/(DF\uparrow)$

$(AB+C*)/(CDF\uparrow)$

$AB+C*CDF\uparrow/$

2009

Q convert from infix to postfix to following expression

$A+B*C/D$

prefix

postfix

2016

NOTES

DATE

2010 Translate following infix expression into it's equivalent prefix expression

$A+(B*C)/(C-(D*B))$

2012 Q3 Translate each of the following infix expression to postfix form

(a) $(A+B*C-D)/(E*F)$

(b) $A+(((B-C*(D-E)+F)/G)*(H-I))$

Prefix

$A+(((B-C*(-DE)+E)/G)*(-HI))$

$=A+(((B-(*C-DE)+F)/G)*(-HI))$

$=A+(((B-(+*C-DEF))/G)*(-HI))$

$=A+((-B+*C-DEF)/G)*(-HI))$

$=A+((/--B+*C-DEFG)*(-HI))$

$=A+(*/-B+*C-DEFG-HI)$

$=+A*/-B+*C-DEFG-HI$

NOTES

DATE

$A+(((B-C*(D-E)+F)/G)*(H-I))$

$=A+(((B-C*(DE-)+F)/G)*(HI-))$

$=A+(((B-(CDE-*)+F)/G)*(HI-))$

$=A+(((B-(CDE-*F+))/G)*(HI-))$

$=A+((((BCDE-*F+-)/G)*(HI-))$

$=A+(((BCDE-*F+-G/)*(HI-))$

$=A+(BCDE-*F+-G/HI-*)$

$=ABCDE-*F+-G/HI-*+$

NOT AND OR

NOTES:

7) NOT A OR NOT B NOT C

NOT A OR NOT B NOT C DATE

postfix

(A NOT) OR (B NOT) AND (C NOT)

= (A NOT) OR (B NOT C NOT AND)

= A NOT B NOT C NOT AND OR

prefix

(NOT A) OR (NOT B) AND (NOT C)

= (NOT A) OR (AND NOT B NOT C)

= OR NOT A AND NOT B NOT C

8) (True && False) || ! (False || True)

postfix

(&& TF) || ! (|| FT)

= (&& TF) || (! || FT)

= || && TF ! || FT

postfix

= (TF &&) || ! (FT ||)

= (TF &&) || (FT || !)

= TF && FT || ! ||

2017/2015

NOTES

Q Transform each of the following expressions to infix form

(i) + - ABC

(ii) + A - BC

(iii) + - / AC * D ↑ FFG

NOTES

DATE

(ii) $+A-B\,C$

$+A\,(B-C)$

$A+(B-C)$

(iii) $+-/AC*DTEFG$

$+-(A/C)*D(E\uparrow F)G$

$+-((A/C)(D*E\uparrow F)G$

$+((A/C)-(D*E\uparrow F))G$

$((A/C)-(D*E\uparrow F))+G$

Conversion

NOTES

DATE

Tabular Method

Q Obtain the postfix notation of the following infix notation of expression. showing the contents of the stack and postfix expression formed after each step of conversion.

2011 (II) A * B + (C - D) / F

2013 (III) A * B + (C - D) / (E * F)

(IV) (A + B) * C / D $ F

(V) NOT A OR NOT B NOT C

(VI) (True && False) || ! (False || True)