DRT (4)

(4) Conversion of Expressions (22)

* Conversion of Infix to postfix (normal format to postfix. post means after.]

Ex:- ~~((A+B)/(C*D))~~

(A+B)/(C*D)

operators should come after the operand.

(A + B)/(C * D)

Highest priority ↑
* /
+ -
Lowest

Reading character	Stack	postfix
(	(	
A	(	A
+	(+	A
B	(+	AB
)	(+)	AB+
/	/	AB+
(	(/	AB+
C	(/	AB+C
*	(/*	AB+C
D	(/*	AB+CD
)	(/*)	AB+CD*/

∴ O/P is AB+CD*/

One more example :- A+(B*C-(D/E-F)*G)*H

chara	stak	postfix
A		A
+	+	A
(	+(	A
B	+(	AB
*	+(*	AB
C	+(*	ABC

	+ (-	ABC *
(	+ (- (	ABC *
D	+ (- (	ABC * D
/	+ (- (/	ABC * D
E	+ (- (/	ABC * DE
-	+ (- (-	ABC * DE /
F	+ (- (-	ABC * DE / F
)	+ (- (-)	ABC * DE / F -
*	+ (- *	ABC * DE / F -
G	+ (- *	ABC * DE / F - G
)	+ (- *)	ABC * DE / F - G * -
*	+ *	ABC * DE / F - G * -
H	+ *	ABC * DE / F - G * - H

Final Expn is postfix. ABC*DE/F-G*-H**+

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

postfix to infix

Scanned with CamScanner

Convert the following postfix exp to infix Expn. (2)

AB - DE + F * /

Reading of post fix	Stack top.	Expn.
① A	A	A
② B	-B	B / A
③ -	(A - B)	A-B
④ D	D	D / A-B
⑤ E	E	E / D / A-B
⑥ +	(D+E)	D+E / A-B
⑦ F	F	F / D+E / A-B
⑧ *	(D+E) * F	D+E * F / A-B

∴ ~~The final expression is~~ (A-B) / (D+E) * F

9) /

∴ Final Infix Exp is (A-B) / (D+E) * F

* Conversion of prefix Exp to infix Exp^n

Ex :- + - * A B / C D E

E D C / B A * - +

we can't place operators on the stack ∴ we convert the prefix Exp^n.

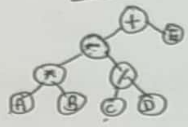
	Reading of prefix	Exp^n stack top	stack top
1	E	E	E
2	D	D	D / E
3	C	C	C / D / E
4	/	(C/D)	C/D / E
5.	B	B	B / C/D / E
6	A	A	A / B / C/D / E
7	*	(A*B)	(A*B) / C/D / E
8	-	[(A*B) - C/D]	(A*B) - C/D / E
9.	+	(A*B - C/D) + E	[A*B - C/D] + E

If operator comes pop 2 top most operands.

Therefore the infix expression is

$$= [(A*B) - (C/D)] + E$$

* Convert the expression $(A * B) - (C/D) + E$ (Infix Exp^n) to prefix Exp^n.



AB*CD/-E+

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

$(*AB) - (/CD) + E$

$T_1 = *AB$

$T_2 = /CD$

$\Rightarrow [T_1 - T_2] + E$

$\Rightarrow (-T_1 T_2) + E$

$T_3 = -T_1 T_2$

$\Rightarrow T_3 + E$

$\Rightarrow + T_3 E$

$\Rightarrow + - (T_1 T_2) E$

+ - * A B / C D E

Final answer, is

+ - * A B / C D E

*/ highest priority

Infix to postfix

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

Char	Stack	Exp^n
(	(	
A	(	A
*	(*	A
B	(*	AB
)	(*)	AB*
-	-	
(	-(	
C	-(	AB*C
/	-(/	AB*C
D	-(/)	AB*C/
+	-+	
E		AB*CD/E+

② Convert infix to prefix. given $(A+B)/C * D - E$

$T_1 = +AB$

$\Rightarrow [+AB] / C * D - E$

$\Rightarrow [T_1 / C] * D - E$

$/T_1 C = T_2$

$\Rightarrow /T_1 C * D - E$

$\Rightarrow [T_2 * D] - E$

$T_3 = *T_2 D$

$\Rightarrow *T_2 D - E$

$= T_3 - E$

$= -T_3 E$

Now substitute T_3 T_2 & T_1

$-*T_2 D E$

$-*/T_1 C D E$

$-*/+ABCDE$

Final Exp^n is prefix

- * / + ABCDE

Convert from infix to ~~postfix~~ prefix

Ex:- $(A*B)*+C$

* Reverse the expression we get postfix then Reverse the o/p.
* Reverse the expression $C+(B*A)$.

Reading	Stack	postfix
C	+	C
+	+	C
(	+(	C
B	+(	CB
*	+(*	CB
A	+(*	CBA
)	+(*)	CBA*
		pop +

o/p = CBA*+

Now Reverse it we get prefix Expn.

+*ABC

(2) Ex:- $((A+(B*C-(D/E^{\wedge}F)*G))$

Reverse it $((G*(F^{\wedge}E/D)-C*B)+A)$

Reading	Stack	postfix
(	(	
(	((	G
*	*	G
		G

(	((*(	G
F	((*(	GF
^	((*(^	GF
E	((*(^	GFE
/	((*(/	GFE^
D	((*(/	GFE^D
)	((*(/) remove	GFE^D/
-	((*-	GFE^D/*
C	((-	GFE^D/*C
*	((-*	GFE^D/*C
B	((-*	GFE^D/*CB
)	((-*) pop & remove	GFE^D/*CB*-
+	((+	GFE^D/*CB*-
A	((+	GFE^D/*CB*-A
)	((+)	GFE^D/*CB*-A+

o/p is GFE^D/*CB*-A+ → postfix.

Reverse the o/p to get prefix expression.

+A-*BC*/D^EFG — prefix

Conversion of infix to postfix.

① $(A+B) * C$

$T_1 = AB+$

$\Rightarrow (AB+) * C$

$\Rightarrow T_1 * C$

$\Rightarrow T_1 C *$ Substitute T_1

$= AB+C*$

② $(A+B) / (C-D)$

$T_1 = AB+$

$T_2 = CD-$

$\Rightarrow (AB+) / (CD-)$

$\Rightarrow T_1 / T_2$

$\Rightarrow T_1 T_2 /$

$\Rightarrow AB+CD-/$

③ $A + (B / C) - D$ → highest priority

$A + BC/ - D$ $\quad T_1 = BC/$

$\Rightarrow A + T_1 - D$

$T_2 = AT_1 +$

$\Rightarrow AT_1 + - D$

$\Rightarrow T_2 - D$

$\Rightarrow T_2 D -$

$\Rightarrow AT_1 + D -$

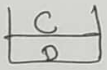
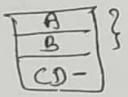
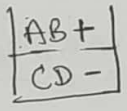
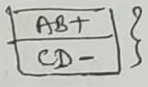
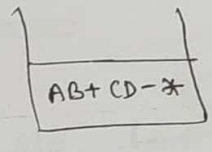
Now substitute T_1, T_2

$\Rightarrow ABC/+D-$

Conversion from prefix to postfix.

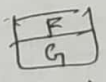
Ex:- * + AB − CD ← start from right to left.

when operator comes pop two operands from the stack

Reading character	Expn	Stack top
D	D	
C	C	
−	CD −	
B	B CD −	
A	AB CD − BA	
+	AB+CD −	
*	AB+CD− *	
		

∴ Final postfix expn is **AB+CD−***

Ex:- 2 +/+A↑BD−EFG ← Take in reverse order

char	Expn	Stack
G	G	
F	F	
E	E	

–	EF–	EF– / G
D	D EF– G	D / EF– / G
B	BDEF– G	B / D / EF– / G
↑	BD↑EF– G	BD↑ / EF– / G
A	ABD↑EF– G	A / BD↑ / EF– / G — ABD↑
+	ABD↑+ EF– G	ABD↑+ / EF– / G
/	ABD↑+ EF– / G	ABD↑+EF–/G / G
+	ABD↑+EF–/G+	

∴ Final o/p is ABD↑+EF–/G+ (postfix Expn)

postfix to Infix :-

A	B / D		E–F / A+(B↑D)
B	(B↑D) / A		G / A+(B↑D)/(E–F)
D	A+(B↑D)	/	
↑		G	
+	F / E / A+(B↑D)	+	A+(B↑D)/(E–F)+G
E			
F			

Conversion from postfix to prefix

Ex:- AB + CD − *

→

Chara	Exp	stack
A	A	A
B	AB	B / A
+	+AB	+AB
C	C+AB	C / +AB
D	DC+AB	D / C / +AB
−	−CD+AB	−CD / +AB
*	*−CD+AB	

−DC
+BA

*−DC+BA

o/p *+AB−CD. prefix exp^n.

D / B / A

D^B / A

E

(D^B)+A

Ex 2:- ABD^ + EF−/G+

Chara	Expn	Stack.
A	A	A
B	AB	B / A
D	D	D / B / A (op1, op2)

(F−E) (D^B)
(D^B+A)
G /
(F−E)/(D^B)+A
G+(F−E)/(D^B)+A
A+(B^D)/(E−F)+G

^ ^ BD A

^ BD
A

↑ B D
op2 op1

+ +A^BD

+A^BD
op2 op1

E E
+A^BD

F F
E
+A^BD

− − EF
+A^BD
opr 1
opr 2

/ /+A^BD−EF

G G
/+A^BD−EF

+ +/+A^BD−EFG

Final o/p ⇒ +/+A^BD−EFG

Scanned with CamScanner

Algo to convert infix to postfix exp.

Q1) $(A+B)/(C*D)$

ch	stack	Postfix
(	(	
A		A
+	(+	
B		AB
)	(+)	AB+
/	/	AB+
(	/(	AB+
C		AB+C
*	/(*	
D		AB+CD
)	/(*)	AB+CD*/

postfix exp.

Q2) $A+(B*C-(D/E-F)*G)*H$

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

Final Answer: ABC*DE/F-G*-H*+

How to convert infix expression to prefix

Q1. $(A*B)+C$

Reverse → postfix → Reverse o/p

$C+(B*A)$

ch	Stack	Postfix
C		C
+	+	C
(	+(	
B		CB
*	+(*	
A		CBA
)	+(*) →	CBA*
	+	

CBA*+ this is postfix

Reverse it

+*ABC → Prefix Expn

Q2:- $A+(B*C-(D/E\wedge F)*G)$

Reverse:- $(G*(F\wedge E/D)-C*B)+A$

ch	Stack	postfix
(	(	
G	(	G
*	(*	G
(	(*(	G
F	(*(	GF
^	(*(^	GF
E	(*(^	GFE
/	(*(/ →	GFE^
D	(*(/	GFE^D
)	(*(/) →	GFE^D/
-	(*- →	GFE^D/*
C		GFE^D/*C
*	(-*	GFE^D/*CB
B		
)	(-*) →	GFE^D/*CB*-
+	+?	
A		GFE^D/*CB*-A+

Reverse it

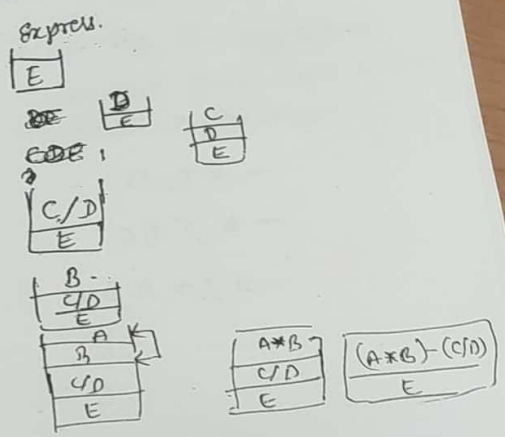
+A-*BC*/D^EFG

Conversion of prefix expression to infix Expression

(i) +-*AB/CDE

Reverse Prefix expression EDC/BA*-+

Prefix	stack top
E	E
D	D
C	C
/	C/D
B	B
A	A
*	A*B
-	(A*B)-(C/D)
+	(A*B)-(C/D)+E



1. Read the prefix expression in reverse order (from right to left)
2. If the symbol is an operand, then push it onto the stack.
3. If the symbol is an operator then pop two operands from the stack.
   Create a string by concatenating the two operands & the operator before them.
   String = operator1 + operator2.
   Push the result into the stack
4. Repeat the steps.

Infix to prefix $(A+B)/C*D-F$

$T_1 = +AB$

$T_2 = *CD$

$T_1/C*D-F$

T_2*D-F

T_3-F

$T_2 = /T_1C$

$T_3 = *T_2D$

$-T_3F$

$-T_3F$

$-*T_2D$

$-*/T_1C$

$-*/+ABC$

Infix, Prefix and Postfix conversions ①

a) Infix to Prefix

(i) $a * (b+c) / e - f$

i Reverse the Expression
ii convert from infix to postf
iii reverse the Expn

$f - e / (c+b) * a$

Symbol	stack	o/p
f	-	f
-	-	f
e	-	fe
/	-/	fe
(	-/(	fe
c	-/(	fec
+	-/(+	fec
b	-/(+	fecb
)	-/(+)	fecb+
*	-/*	fecb+
a	-/*	fecb+a
		fecb+a*/-

-/*a+bcf

b) Infix to postfix (A+B)*C-(D-E)/(F+G)

Symbol	Stack	o/p
(	(	
A	(	A
+	(+	A
B	(+	AB
)	(+)	AB+
*	*	AB+
C	*	AB+C
-	-*	AB+C*
(	-(	AB+C*
D	-(	AB+C*D
-	-(-	AB+C*D
E	(-	AB+C*DE
)	-(-)	AB+C*DE-
/	-/	AB+C*DE-
(	-/(	AB+C*DE-
F	-/(	AB+C*DE-F
+	-/(+	AB+C*DE-F
G	-/(+	AB+C*DE-FG
)	-/(+)	AB+C*DE-FG+
	-/	AB+C*DE-FG+/-

Any :- AB+C*DE-FG+-/

Convert postfix (i) AB-DE+F*/

Symbol	Stack top
A	A
B	B
-	-

Convert Postfix Expn to Infix Expn

(i) AB-DE+F*/

Symbol	Stack top	Expression
A	A	A
B	B	B A
-	(A-B)	A-B
D	D	D A-B
E	E	E D A-B
+	D+E	D+E A-B
F	F	F D+E A-B
*	((D+E)*F)	(D+E)*F A-B
/	(D+E)*F/(A-B)	

(A-B)/(D+E)*F

Ans :- (A-B)/(D+E)*F

Prefix Exp^n to Infix Exp^n.

(a) + − * AB/CDE * Make it reverse EDC/BA*−

Symbol	Stack top	Exp^n
E	E	E
D	D	D, E
C	C	C, D, E
/	D/C	D/C, E
B	B	B, D/C, E
A	A	A, B, D/C, E
*	(A * B)	B*A, D/C, E
−	D/C − (B*A)	D/C − (B*A), E
+	E + D/C − B*A	E+D/C − B*A

Reverse it

Ans is (A * B) − (C/D) + E

Scanned with CamScanner

Conversion from Prefix to postfix. Two methods (5)

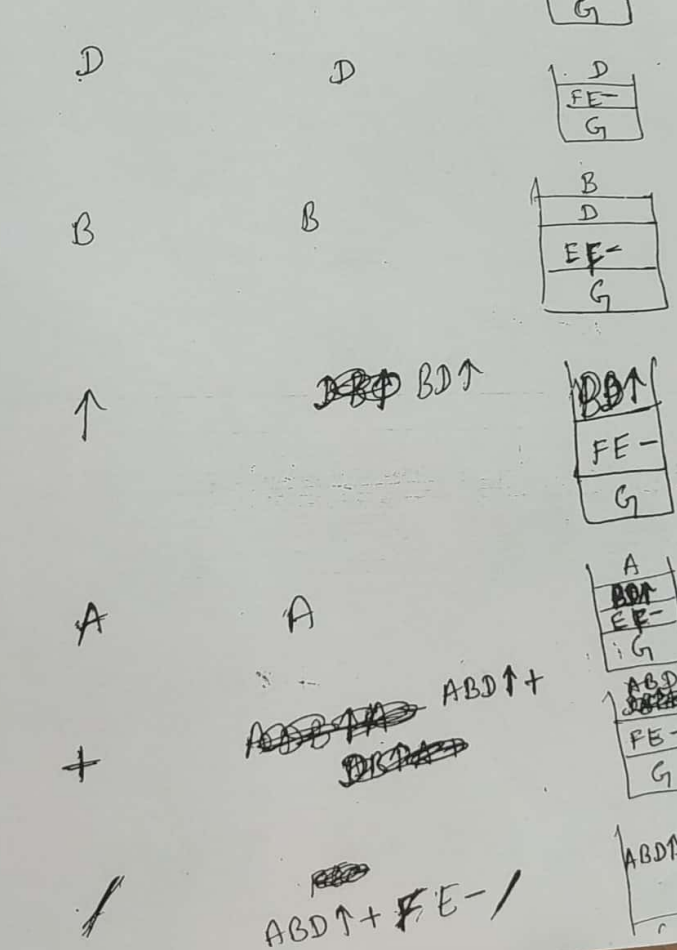
(i) Prefix to infix then infix to postfix

(ii) Direct method.

Ex:- (i) +/+A*BD-EFG

Reverse it GFE-DB↑A+/+

	Symbol	Stack top	Exp.
①	G	G	+ ABD↑+EF-/G+
②	F	F	
③	E	E	Final Answer: ABD↑EF-/G+
	-	EF-	
	D	D	
	B	B	
	↑	BD↑	
	A	A	
	+	ABD↑+	
	/	ABD↑+EF-/	



Assignment

Convert the Expression $(A+B/C*(D+E)-F)$ to prefix & postfix Expressions.

Ans :- ABC/DE+*+F- postfix.

1) postfix to prefix

AB-DE+F*/

Sy	Stack	Exp
A	A	A
B	B, A	B
-	-BA	-BA
D	D, -BA	D
E	E, D, -BA	E
+	+ED, -BA	+ED
F	F, +ED, -BA	F
*	*F+ED	*F+ED
/	/*F+ED-BA	