1. **Consider the following arithmetic Infix Expression. Convert given expression into Prefix (Polish) Notation. Write algorithm and Symbol Table which shows status of stack.**

Q: A + (B * C - (D / E ^ F) * G) * H

Algorithm of Infix to Prefix

Step 1. Reverse the given expression.
Step 2. Scan from right to left and repeat steps 3, 4,5,6 until the stack is empty.
Step 3. If Expression comes, then Push in P.
Step 4. If '(' parenthesis encounter, then Push it on stack.
Step 5. If ')' comes, then every operator from stack push in expression P till ')' is encountered.
Remove ')'. [Do not add ')' to P.
[End of If Structure]
Step 6. If operator comes, then

a. Check precedence of scanned operator with operator TOS.If precedence of TOS is same or greater than scanned operator, repeatedly Pop from stack and add to P expression.
b. Push it on stack.

[End of If Structure]
Step 7. Reverse the postfix expression.
Step 8. Exit.

Symbol Table

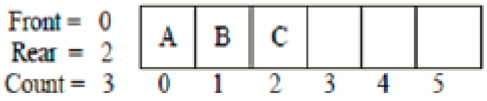
Q: A + (B * C - (D / E ^ F) * G) * H

S.no	Symbol Scanned	Stack	Output Prefix (P)
1.	H	)	H
2.	*	*)	H
3.	)	)*)	H
4.	G	)*)	GH
5.	*	*)*)	GH
6.	)	)*)*)	GH

7.	F	)*)*)	FGH
8.	^	^)*)*)	FGH
9.	E	^)*)*)	EFGH
10.	/	/)*)*)	^EFGH
11.	D	/)*)*)	D^EFGH
12.	(	*)*)	/D^EFGH
13.	-	-)*)	*/D^EFGH
14.	C	-)*)	C*/D^EFGH
15.	*	*-)*)	C*/D^EFGH
16.	B	*-)*)	BC*/D^EFGH
17.	(	*)	-* BC*/D^EFGH
18.	+	+)	*-* BC*/D^EFGH
19.	A	+)	A*-*BC*/D^EFGH
20.	(		+A*-*BC*/D^EFGH

P: + A * - * B C * / D ^ E F G H

2. Make Queue using Array[6] and apply following functions:



dequeue( ) , enqueue(D) , enqueue(E) , enqueue(F), dequeue( ) , dequeue( ), enqueue(G), dequeue( ), dequeue( ), enqueue(H), enqueue(I), dequeue( ), dequeue( ), dequeue( ), dequeue( )

<u>SOLUTION:</u>

STEP#1 Front = 0, Rear = 2, count = 3

A	B	C			
0	1	2	3	4	5

STEP#2 Front = 1, Rear = 2, count = 2
dequeue() (A is deleted from Front)

	B	C			
0	1	2	3	4	5

STEP#3 Front = 1, Rear = 5, count = 5
enqueue() (D, E and F are inserted in stack from Rear)

	B	C	D	E	F
0	1	2	3	4	5

STEP#4 Front = 3, Rear = 5, count = 3
dequeue() (B and C deleted from Front)

			D	E	F
0	1	2	3	4	5

STEP#5 Front = 3, Rear = 0, count = 4
enqueue() (G is inserted in stack as Rear)

G			D	E	F
0	1	2	3	4	5

STEP#6 Front = 5, Rear = 0, count = 2
dequeue() (D and E are deleted from Front)

G					F
0	1	2	3	4	5

STEP#7 Front = 5, Rear = 2, count = 4
enqueue() (H and I are inserted from Rear)

G	H	I			F
0	1	2	3	4	5

STEP#8 Front = -1, Rear = -1, count = 0
dequeue() (F, G, H and I are deleted from Front)

0	1	2	3	4	5

Stack is Empty.