Expression Conversion:

1. Infix to postfix - Algorithm: Scan from left to right

Step 1: Declare a stack of operators. (Assume that we are using binary operators).

Step 2: Check if the scan symbol is operand, display it as the output.

Step 3: Check, If it is an operator then,

a. Check, if the operator stack is empty, push it to the operator stack.

b. Check, if the operator stack is non-empty, compare the precedence of the operator and the top of stack.

i. If the operator's precedence is greater than to the precedence of the stack top of the operator stack, then push the character to the operator stack.

ii. If the incoming operator has equal precedence with the top of the stack, use association. If the association is left to right, pop and print the top of the stack and then push the incoming operator. If the association is right to left, push the incoming operator.

iii. Otherwise, pop the elements from the stack until the character's precedence is less or the stack is empty.

c. If the character is left parenthesis, "(", push it into the o operator stack.

d. If the character is right parenthesis, ")", then pop until "(" is encountered in the operator stack.

Example

Input expression: `(k+l-m*n+(o^p)*w/u/v*t+q)`

Operator Stack	Postfix Expression	Top of Stack
((		2
((	k	2
((+	k	3
((+	kl	3
((-	kl+	3
((-	kl+m	3
((-*	kl+m	4
((-*	kl+mn	4
((+	kl+mn*-	3

```
                ((+( |              kl+mn*- |                    4
                ((+( |             kl+mn*-o |                    4
               ((+(^ |             kl+mn*-o |                    5
               ((+(^ |            kl+mn*-op |                    5
                 ((+ |           kl+mn*-op^ |                    3
                ((+* |           kl+mn*-op^ |                    4
                ((+* |          kl+mn*-op^w |                    4
                ((+/ |         kl+mn*-op^w* |                    4
                ((+/ |        kl+mn*-op^w*u |                    4
                ((+/ |       kl+mn*-op^w*u/ |                    4
                ((+/ |      kl+mn*-op^w*u/v |                    4
                ((+* |     kl+mn*-op^w*u/v/ |                    4
                ((+* |    kl+mn*-op^w*u/v/t |                    4
                 ((+ |  kl+mn*-op^w*u/v/t*+ |                    3
                 ((+ | kl+mn*-op^w*u/v/t*+q |                    3
                   ( | kl+mn*-op^w*u/v/t*+q+ |                    1
-------------------------------------------------------------------------
Final Postfix Expression: kl+mn*-op^w*u/v/t*+q+
-------------------------------------------------------------------------
```

2. Infix to Prefix -

a. Perform the reverse of the given input expression, and change the order of parenthesis, i.e. (to) and vice versa.
b. Obtain the postfix expression.
c. Now, reverse the output postfix, then the final expression would be the prefix result.

Example:
Input expression: **(k+l-m*n+(o^p)*w/u/v*t+q)**

a. Reverse and change the order of parenthesis.
b. Obtain the postfix expression.

```
Operator Stack |   Postfix Expression |         Top of Stack
-------------------------------------------------------------------
                (( |                     |                    2
                (( |                   q |                    2
                ((+ |                   q |                    3
                ((+ |                  qt |                    3
               ((+* |                  qt |                    4
               ((+* |                 qtv |                    4
               ((+/ |                qtv* |                    4
               ((+/ |               qtv*u |                    4
               ((+/ |              qtv*u/ |                    4
               ((+/ |             qtv*u/w |                    4
               ((+* |            qtv*u/w/ |                    4
              ((+*( |            qtv*u/w/ |                    5
              ((+*( |           qtv*u/w/p |                    5
             ((+*(^ |           qtv*u/w/p |                    6
             ((+*(^ |          qtv*u/w/po |                    6
               ((+* |         qtv*u/w/po^ |                    4
                ((+ |       qtv*u/w/po^*+ |                    3
```

```
                 ((+ |       qtv*u/w/po^*+n |                    3
                ((+* |       qtv*u/w/po^*+n |                    4
                ((+* |      qtv*u/w/po^*+nm |                    4
                 ((- |    qtv*u/w/po^*+nm*+ |                    3
                 ((- |   qtv*u/w/po^*+nm*+l |                    3
                 ((+ |  qtv*u/w/po^*+nm*+l- |                    3
                 ((+ | qtv*u/w/po^*+nm*+l-k |                    3
                   ( | qtv*u/w/po^*+nm*+l-k+ |                     1
-------------------------------------------------------------------
Final Postfix Expression: qtv*u/w/po^*+nm*+l-k+
-------------------------------------------------------------------
```

c. Reverse the final postfix expression to prefix one.
+k-l+*mn+*^op/w/u*vtq