**Kathmandu University**

Department of Computer Science and Engineering

Dhulikhel



Mini Project

Of

COMP 202

(Data Structure and Algorithm)

Submitted by: Submitted to:

Sabal Katuwal Miss Rajani Chulyadyo

CE-II-I DoCSE

**Project Report**

Objective: Write a program to convert prefix to postfix also find its time complexity.

Algorithm: Prefix To Postfix Conversion

Input: Prefix expression

Output: Postfix expression

Steps:

1. Scan PREFIX expression from RIGHT to LEFT. -------O(n)
2. IF the incoming symbol is a OPERAND, PUSH it onto the Stack ------O(1)
3. IF the incoming symbol is a OPERATOR, POP 2 OPERANDs from the Stack, ADD this incoming OPERATOR in before the 2 OPERANDs & PUSH this whole new expression string back into the Stack. ----------O(1)
4. At the end POP and PRINT the full POSTFIX expression from the Stack. ------O(1)

Overall complexity : O(1)

**Test program :**

Prefix: ++A\*BCD

Push ‘D’ Push ‘C’ Push ‘B’ ‘ \* ’is operator so, Push A

temp = B+D+’\*’ = BD\*

Push BD\*

B A

C C BC\* BC\*

D D D D D

‘+’ is operator so, ‘+’ is operator so,

Temp = ‘A’ + ‘BC\*’ + ’+’ = ABC\*+ Temp = ‘ABC\*+’ + ‘D’ + ‘+’ = ABC\*+D+

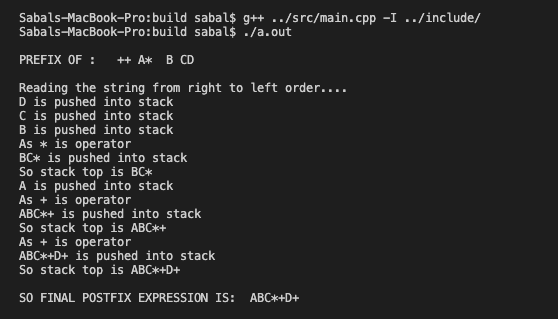
Push ABC\*+ Push ABC\*+D+

ABC\*+ ABC\*+D+

D

Now, top(stack) = ABC\*+D+

Therefore postfix is = ABC\*+D+



**Time complexity:**

The time complexity of the program is O(n) where n is the size of Prefix Expression.

Each stack operation is performed in O(1)

i.e. isEmpty() = O(1)

isFull() = O(1)

pop()= O(1)

peek()= O(1)

Overall stack operation is O(1)

Similarly,

For PreToPost class the loop for runs from n-1 to 0. Because of this, its performance is O(n).

Hence the overall performance becomes O(n).