NAME: RAMYA RAMESH
USN: 1BM19CH038

Lab3: Infix to Postfix

WAP to convert a given valid parenthesized infix arithmetic expression to postfix expression. The expression consists of single character operands and the binary operators + (plus), - (minus), * (multiply) and / (divide).

Pseudocode:

```
InfixtoPostfix (exp)
{
    create a stack S
    for i<0 to length(exp) - 1
    {
        if exp[i] is operand
            res ← res + exp[i]
        elseif exp[i] is operator
        while (!Sempty() && HasHigherPre(S.top(),
        exp[i])
            {
                res ← res + s.top()
                S.pop()
            }
        S.push(exp[i])
    else if IsOpeningParentheses(exp[i])
            S.push(exp[i])
    else if IsClosingParentheses(exp[i])
    {
        while (!S.empty() && !IsOpeningParenthes
        (S.top()))
            {
                res ← res + s.top()
```

```
            s.pop()
        }
    s.pop()
    }
}
while (!s.empty())
{
    res ← res + s.top()
    s.pop()
}
return res
}
```

Algorithm

1.) Push "(" onto Stack and add ")" to the end of X.
2.) Scan X from left to right and repeat Step 3 to 6 for each element of X until Stack is empty.
3.) If an operand is encountered, add it to Y.
4.) If a left parenthesis is encountered, push it into Stack.
5.) If an operator is encountered, then:
    (i) Repeatedly pop from Stack and add to Y each operator (on top of stack) which has the same precedence as or higher precedence than operator.
    (ii) Add operator to Stack
    [End of If]
6.) If a right parenthesis is encountered then:
    (i) Repeatedly pop from Stack and add to Y each operator (on the top of Stack) until a left parenthesis is encountered
    (ii) Remove the left Parenthesis
    [End of If]
    [End of If]
7.) End