Karthik Deepak
1BM19CS200 [D-2] 5-10-2020
D Section

Infix to Postfix Pseudo Code

```
Infix to Postfix (exp)
{
  Create a Stack S
  for i<-0 to length (exp)-1
  {
     if exp[i] is operands
          res <- res + exp[i]
     else if exp[i] is operator
    while (!S.empty () && HasHigherPre (S.top (). exp[i]))
      {
         res <- res + S.top()
         S.pop()
      }
     S.push(exp[i])
    else if IsOpening Parenthesis [exp[i])
          S.push (exp [i])
    else if IsClosing Parenthesis (exp[i])
      {
        while (!S .empty() && ! Is Opening Parentheses [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 unit the Stack is empty

3 If an operand is encountered, add it to Y
4 If a left parenthesis is encountered, push it onto Stack
5 If an operaton is encountered then

      Repeatedly pop from stack and add to Y each
      operator (on top of Stack) which has the
      same procedance as or higher precedence than operator
      Add operator on to Stack

6 If a right parenthesis is encounted

  i] Repeatedly pop from stack and adds to Y each
     operator Con the top of stack until the left
     parenthesis is encountered
        end of 'y
        end of 'y
     End
```