Pseudocode to convert infix expression to postfix

```
Input: Infix expression (infix)

Output: Postfix expression (postfix)

Initialize postfix ← “”, stack ← empty stack

Get next symbol from infix
    If symbol is an operand
        Append symbol to postfix
    If symbol is ‘(‘
        Push symbol  on the stack
    If symbol is ‘)’
        While top of stack is not ‘(‘
            Pop operator from stack
            Append to postfix
        Pop the ‘(‘ from the stack and discard it
    If symbol is an operator
        While top of stack is not ‘(‘
            Pop operator from stack
            If popped operator has higher precedence than symbol
                Append operator to postfix
            Else
                Push operator back on to stack
                Push symbol on to stack
                Break
            Push symbol on to stack
    If symbol is a ‘;’
        Exit from the loop

While stack is not empty
    Pop operator from stack
    Append to postfix
```

Pseudocode to evaluate a postfix expression

```
Input: Postfix expression (postfix)

Output: Evaluation of the expression (result)

Initialize result ← 0, stack ← empty stack

Get next symbol from postfix
    If symbol is a number
        Push symbol  on the stack
    If symbol is an operator
        Pop n1 from stack
        Pop n2 from stack
        Compute  res = n1 operator n2
        Push res on to stack

While stack is not empty
    Pop result from stack
```