**Submitted By:- Ravi Pandey (075BCT065)**

Write an algorithm for the following:

1. Perform push and pop operations in stack.

A)Push Operation

The algorithm to perform the push operation in stack is given below:

Push(Stack, MaxN, Data,Top)

If Top=N-1

Print “Stack Overflow!” and return null

Set Top=Top+1

Stack[Top]=Data

Return

B)Pop Operation

The algorithm to perform the pop operation in stack is given below:

Pop(Stack, Item, Top)

If TOP= -1 //stack is empty

Print “Underflow!” and return null

Set Item=Stack[Top]

Set Top=Top-1

Return

2. Convert :

a. Infix expression to postfix expression

1. START
2. Scan the infix expression form left to right
3. If the scanned character is an operand, then add it to postfix expression
4. The scanned character is an operator then:
5. If the operator currently at the top of the stack has lower precedence than the operator read or the stack is empty then push the operator to the stack
6. Else pop the stack and add the popped operator to postfix expression and push the read operator to the stack
7. If stack is not empty after reading all characters from infix expression then pop operators from stack to postfix expression until the stack is empty
8. Display the resulting postfix expression
9. END

b. Infix expression to prefix expression

1. START
2. Reverse the given infix expression
3. Interchange all ‘(‘ into ‘)’ and vice-versa.
4. Convert the obtained expression into postfix expression
5. Reverse the obtained expression so that it becomes the required prefix expression.
6. END

3. Evaluate the postfix expression

1. START
2. Scan each character of the postfix expression from left to right
3. If operand is encountered, push it onto Stack

[End If]

1. If operator is encountered
2. A -> Top element and pop
3. B-> Next to Top element and pop
4. Evaluate B operator A
5. push result of B operator A onto Stack
6. Set result = stack[top]
7. END

4. Check the paired parenthesis in mathematical expression.

1. START
2. Scan each character of the expression from left to right
3. If left parenthesis is encountered, push it onto Stack
4. If right parenthesis is encountered
5. And stack is not empty i.e it contains one left parenthesis then pop it out.
6. And stack is empty then print “Unbalanced Paranthesis” and exit
7. If stack is empty after scanning all the characters then

Print “Balanced parenthesis”

1. END