TUTORIAL NO. – 9

**STACK APPLICATION**

**PROBLEM STATEMENT /DEFINITION:-**

Implement C++ program for expression conversion as infix to postfix and its evaluation using stack based on given conditions

1. Operands and operator, both must be single character.
2. Input Postfix expression must be in a desired format.
3. Only '+', '-', '\*' and '/ ' operators are expected.

**OBJECTIVE:-**

* To understand concept of stack.
* To study the expression conversion as stack application.

**OUTCOME:-**

* To implement primitive operations on stack.
* To make use of stack for converting infix expression to postfix.

**S/W PACKAGES AND HARDWARE APPARATUS USED:-**

Operating Systems (64-Bit) 64-BIT Fedora 17 or latest 64-BIT Update of Equivalent Open source OS or latest 64-BIT Version and update of Microsoft Windows 7 Operating System onwards Programming Tools (64-Bit) Latest Open source update of Eclipse Programming frame work, TC++.

**REFERENCES:-**

* + C++ by B. Stroustrup
  + Fundamental of Data Structure in C++

**STEPS:-**

Refer to algorithm

**Theory:-**

Consider a simple expression: A + B .This notation is called Infix Notation.

A + B in Postfix notation is A B +

As you might have noticed, in this form the operator is placed after the operands (Hence the name 'post'). Postfix is also known as Reverse Polish Notation. Similarly for Infix, the operator is placed inside the operands. Likewise the equivalent expression in prefix would be + A B, where the operator precedes the operands.

**Algorithm:-**

1. Push left parenthesis onto STACK and add right parenthesis at the end of Q.

2. Scan Q from left to right and repeat step 3 to 6 for each element of Q until the STACK is empty.

3. If an operand is encountered add it to P.

4. If a left parenthesis is encountered push it onto the STACK.

5. If an operator is encountered, then

* Repeatedly pop from STACK and add to P each operator which has same precedence as or higher precedence than the operator encountered.
* Push the encountered operator onto the STACK.

6. If a right parenthesis is encountered, then

* Repeatedly pop from the STACK and add to P each operator until a left parenthesis is encountered.
* Remove the left parenthesis; do not add it to P.

7. Exit

Complexity:

Time complexity = O(n).