LAB SESSION 2: STACK DATA STRUCTURE

AIM: To implement expression converter and evaluator using Stack Data Structure.

PROBLEM DEFINITION:

1. Develop a C program to implement the following:
    - Convert infix to postfix
    - Convert infix to prefix
    - Evaluate a postfix expression
    - Evaluate a prefix expression

THEORY: A stack is a data structure that follows the LIFO (last in first out) principle. In this data structure we have a TOP pointer which refers to the element at top of the stack. To retrieve or delete data from the stack we have only one position that is the TOP position.

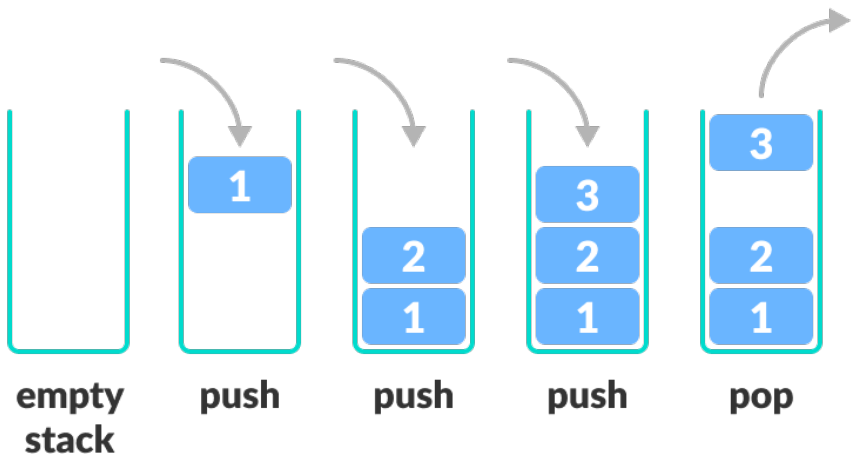

Fig. 2.1: Basic Working of Stack Data Structure

There are some basic operations that allow us to perform different actions on a stack.

- Push: Insert an element to the top of a stack
- Pop: Delete an element from the top of a stack
- IsEmpty: Check if the stack is empty
- IsFull: Check if the stack is full
- Peek: Get the value of the top element without removing it

Working of Stack Data Structure

The operations work as follows:

- A pointer/location called TOP is used to keep track of the top element in the stack.
- When initializing the stack, we set TOP value to -1 so that we can check if the stack is empty by comparing TOP == -1 (for array implementation or TOP = NULL for LL implementation)
- On pushing an element, we increase the value of TOP and place the new element in the position pointed to by TOP.
- On popping an element, we return the element pointed to by TOP and reduce its value.
- Before pushing, we check if the stack is already full
- Before popping, we check if the stack is already empty