Indian Institute of Technology (ISM), Dhanbad
Data Structure Lab (NCSC104)

Assignment-3

1. **Theoretical Description**

   A stack is a linear data structure that follows the Last In, First Out (LIFO) principle. Stacks are analogous to a stack of plates, where the last plate placed on top is the first one to be removed.

   Basic Operations on a Stack

   1. **Push(x)**: Adds an element x to the top of the stack.
   2. **Pop()**: Removes and returns the top element from the stack.
   3. **Peek()/Top()**: Returns the top element without removing it.
   4. **isEmpty()**: Checks if the stack is empty.
   5. **isFull()**: Checks if the stack is full.

2. **The Problem: Implementing a Stack-Based Expression Evaluator**

   You are tasked with creating a **Mathematical Expression Evaluator** for a simple calculator application. The application should:

   1. **Validate the Expression (Parenthesis Matching)**
      a. Ensure that the input expression has balanced parentheses before proceeding with evaluation.
      b. Examples:
         i. Input: (2 + 3) * (5 - 2) → Output: Balanced
         ii. Input: (2 + 3 * (5 - 2) → Output: Not Balanced
   2. **Infix to Postfix Conversion**
      a. Convert the input infix expression to postfix format using stack operations.
      b. Consider operator precedence and associativity rules.
      c. Examples:
         i. Input: (2 + 3) * 5 → Output: 2 3 + 5 *
         ii. Input: 10 + (6 * 2) → Output: 10 6 2 * +
   3. **Postfix Expression Evaluation**
      a. Evaluate the postfix expression using a stack.
      b. Use stack operations to perform the calculations step-by-step.
      c. Examples:
         i. Input: 2 3 + 5 * → Output: 25
         ii. Input: 10 6 2 * + → Output: 22

4. **Interactive User Input**
    a. Allow the user to input a mathematical expression.
    b. Display intermediate steps, including:
        i. Validity of the expression.
        ii. Postfix conversion result.
        iii. Final evaluated result.