## **Instructions**

**Objective**:  
Your task is to create a Java program that reads a Boolean expression in infix notation, converts it to postfix notation, and generates a truth table for the given expression. To handle the expression parsing, you will implement a stack using a singly linked list.

**Requirements**:

1. **Expression Parsing**:
   * The program should take a Boolean expression in infix notation (e.g., A + B \* C) and convert it to postfix notation (e.g., AB+C\*).
   * Use the **Shunting Yard Algorithm** for conversion from infix to postfix.
2. **Truth Table Generation**:
   * Based on the variables in the Boolean expression, generate a truth table.
   * For example, if the expression is A + B, the truth table will show all possible combinations of A and B along with the output of the expression.
3. **Stack Implementation**:
   * Implement a stack using a singly linked list to manage operators and operands during parsing.
   * The stack should support basic operations: push, pop, and peek.
4. **Sample Output**:
   * Given the input A + B, the program should output the following truth table:  
     A | B | A + B  
     --------------  
     T | T | T  
     T | F | T  
     F | T | T  
     F | F | F

**Guidelines**:

* You are required to use a custom stack class with a singly linked list structure, without relying on Java’s built-in Stack or LinkedList classes.
* Ensure your program can handle simple Boolean expressions containing variables (A, B, C, etc.) and operators (+ for OR, \* for AND, ! for NOT).
* Include comments to explain each part of your code.