Postfix to Prefix Conversion Algorithm

1. Initialize an empty stack to store operands (sub-expressions).
2. Scan the postfix expression from left to right.
3. For each character in the postfix expression:
    - **IF** the character is an operand (digit, variable, etc.), push it onto the stack.
    - **IF** the character is an operator:
        - a. Pop the top two operands from the stack.
        - b. Form the prefix expression by concatenating the operator, followed by the two operands (i.e., `operator operand2 operand1`).
        - c. Push the resulting prefix sub-expression back onto the stack.
4. Repeat this process until the entire postfix expression is scanned.
5. At the end, the stack will contain the final prefix expression.

Prefix to Postfix Conversion Algorithm

1. Initialize an empty stack to store operands (sub-expressions).
2. Scan the prefix expression from right to left.
3. For each character in the prefix expression:
    - **IF** the character is an operand (digit, variable, etc.), push it onto the stack.
    - **IF** the character is an operator:
        - a. Pop the top two operands from the stack.
        - b. Form the postfix expression by concatenating the two operands, followed by the operator (i.e., `operand1 operand2 operator`).
        - c. Push the resulting postfix sub-expression back onto the stack.
4. Repeat this process until the entire prefix expression is scanned.
5. At the end, the stack will contain the final postfix expression.