Algorithm To Evaluate Expressions Using Prefix Notation

Initialization:

1. Read a token from the input stream.
    a. If it is an operand, push it into the operand stack.
    b. If it is an operator, push a marker onto operand stack and push the actual operator onto operator stack.

Loop While (there are more input tokens):

1. Read a token.

2. If token is an operator, push operator onto operator stack and push the marker onto operand stack.

3. Else if, the token is an operand, check the top of the operand stack for a marker.

    a. If the marker is on top of the stack, push the read operand onto the operand stack.

    b. Else If – the top of the operand stack contains another operand:

        Loop While (the top of the stack is an operand):

        1. Pop operand from operand stack.

        2. Pop marker from operand stack and the corresponding operator from the operator stack.

        3. Apply the operation on the read token and popped operand and obtain result.

        4. 1 If the top of the stack is another operand, continue Looping (as stated above in Loop While constraint).

        4.2 Else if the top of the stack is an operator or the operand stack is empty, push the result and exit the loop.

4. Go back to step 1 (loop)

Loop is finished:

When the loop is finished (all of the input has been read), the value on top of the operand stack is the result of evaluating an expression using the prefix notation.

Some possible errors:

The operand stack was not empty at the end of algorithm.

The operator stack was not empty at the end of algorithm.

An operator is the last token in the input.

Reading illegal token from the input.