Recall the stack algorithm I gave you that converts an infix expression to a postfix expression. It assumes that negation is denoted by ~ rather than a minus sign.

1. Modify that algorithm so that it works for normal infix expressions, where negation is denoted by a minus sign, without a pre-processing step. Your algorithm must use just one stack and no global variables.

2. Design an algorithm which converts a prefix expression to a postfix expression. Your algorithm must use just one stack and no global variables.

3. Design an algorithm which converts a prefix expression to an infix expression. Your algorithm must use just one stack and no global variables.

4. Does your algorithm for 3. always give a minimal infix expression? (For example, (a*b)+c is not minimal, since it is equivalent to the shorter expression a*b+c.) If not, does there exist an algorithm for this problem which uses just one stack and no global variables, and which always gives a minimal infix expression? Explain your answer.