Algorithm 1: infix2postfix(I, P)

Input: I, Expression in INFIX notation
Output: P, Expression in POSTFIX notation

1 Push '(' onto the STACK, and add ')' to the end of I.
`/* Scan I from left to right.  */`
2 **for** *each element of I untill the STACK is empty* **do**
3 **if** *an operand is encountered* **then**
4 add it to P
5 **else if** *'(' is encountered* **then**
6 push it onto STACK
7 **else if** *an operator* $\otimes$ *is encountered* **then**
   1. Repeatedly pop from STACK and add to P each operator (on the top of STACK) which has same or higher precedence than $\otimes$.
   2. push $\otimes$ onto the STACK
8 **else if** *')' is encountered* **then**
   1. Repeatedly pop from STACK and add to P each operator (on the top of STACK) until a '(' is encountered.
   2. Remove '('
9 **return** P