Algorithm 6.6: POLISH(Q, P)

```
Suppose Q is an arithmetic expression written in infix notation. This algorithm
finds the equivalent postfix expression P.

1. Push "(" onto STACK, and add ")" to the end of Q.
2. Scan Q from left to right and repeat Steps 3 to 6 for each element of Q
   until the STACK is empty:
3.        If an operand is encountered, add it to P.
4.        If a left parenthesis is encountered, push it onto STACK.
5.        If an operator ⊗ is encountered, then:
```

```
            (a) Repeatedly pop from STACK and add to P each operator (on
                the top of STACK) which has the same precedence as or higher
                precedence than ⊗.
            (b) Add ⊗ to STACK.
          [End of If structure.]
    6.    If a right parenthesis is encountered, then:
            (a) Repeatedly pop from STACK and add to P each operator (on
                the top of STACK) until a left parenthesis is encountered.
            (b) Remove the left parenthesis. [Do not add the left parenthesis to
                P.]
          [End of If structure.]
      [End of Step 2 loop.]
    7. Exit.
```

Example 6.7

Consider the following arithmetic infix expression Q:

Q: A + (B * C – (D / E ↑ F) * G) * H

We simulate Algorithm 6.6 to transform Q into its equivalent postfix expression P.

First we push "(" onto STACK, and then we add ")" to the end of Q to obtain:

Q:	A	+	(	B	*	C	–	(	D	/	E	↑	F	)	*
	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)
											G	)	*	H	)
											(16)	(17)	(18)	(19)	(20)

The elements of Q have now been labeled from left to right for easy reference. Figure 6.12 shows the status of STACK and of the string P as each element of Q is scanned. Observe that

(1) Each operand is simply added to P and does not change STACK.

(2) The subtraction operator (–) in row 7 sends * from STACK to P before it (–) is pushed onto STACK.

(3) The right parenthesis in row 14 sends ↑ and then / from STACK to P, and then removes the left parenthesis from the top of STACK.

(4) The right parenthesis in row 20 sends * and then + from STACK to P, and then removes the left parenthesis from the top of STACK.

After Step 20 is executed, the STACK is empty and

P: A B C * D E F ↑ / G * – H * +

which is the required postfix equivalent of Q.

Symbol Scanned	STACK	Expression P
(1) A	(	A
(2) +	(+	A
(3) (	(+ (	A
(4) B	(+ (	A B
(5) *	(+ (*	A B
(6) C	(+ (*	A B C
(7) −	(+ (−	A B C *
(8) (	(+ (− (	A B C *
(9) D	(+ (− (	A B C * D
(10) /	(+ (− (/	A B C * D
(11) E	(+ (− (/	A B C * D E
(12) ↑	(+ (− (/ ↑	A B C * D E
(13) F	(+ (− (/ ↑	A B C * D E F
(14))	(+ (−	A B C * D E F ↑ /
(15) *	(+ (− *	A B C * D E F ↑ /
(16) G	(+ (− *	A B C * D E F ↑ / G
(17))	(+	A B C * D E F ↑ / G * −
(18) *	(+ *	A B C * D E F ↑ / G * −
(19) H	(+ *	A B C * D E F ↑ / G * − H
(20))		A B C * D E F ↑ / G * − H * +