the stack during the computation.

Test Case

Please test your program with Input1 and Input2, and then check the answers with Output1 and Output2.

Listing 6: Infix to Postfix Conversion.

```
Input1：
a + b * c
Output1：
a b c * +
2

Input2：
a / b - c + d * e - a * c
Output2：
a b / c - d e * + a c * -
2
```

7. Postfix expression evaluation

Please finish a program which can perform postfix expression evaluation and count the max top in stack during the process in C. The input is a prefix expression. The operands are lowercase letters「*a*」to「*z*」; the input operators are「*」,「/」,「+」, and「-」. Furthermore, the precedence of the operators is「*」=「/」>「+」=「-」. First, convert the prefix expression to the postfix form. After that, you should use a stack to process postfix form and record the stack size for each step. The output includes two results: (1) the postfix expression, and (2) the stack size for each step.

Test Case

Please test your program with Input, and then check the answers with Output.

Listing 7 : Postfix expression evaluation

```
Input：
- + a * - / b c d e / f g

Output：
a b c / d - e * + f g / -
```

```
6 1 2 3 2 3 2 3 2 1 2 3 2 1
```

8. Circular queue

Please implement a circular queue **using an array** in C. Please program the enqueue(), dequeue(), and display() functions. The length of an array is defined as six; as a result, there are five spaces to store data. In the input file, we define 1 as enqueue() with a data separated by a space; 2 as dequeuer(), and 3 as display().

Note: You must use an array to implement a circular queue; otherwise, no points will be given.

Test Case

Please test your program with Input1 and Input2, and then check the answers with Output1 and Output2.

Listing 8: Circular queue

```
1  Input1:
2  3
3  1 5
4  1 10
5  3
6  2
7  1 15
8  1 20
9  3
10 Output1:
11 The queue is empty.
12 5 10
13 10 15 20
14 Input2:
15 1 3
16 1 5
17 1 7
18 1 9
19 1 11
20 3
21 1 13
22 2
```