3. Invert a singly linked list

Please implement a function to invert a singly linked list in C. The input includes two lines: a sequence of input numbers, and a sequence of partition codes. The number of the input numbers is multiplication by 9 (e.g, 9, 18, 27,…), so that we can divide the numbers into three partitions. We define 1 as the front partition, 2 as the middle partition, and 3 as the rear partition. As the example shown in the following input, there are 9 numbers so each partition has three numbers (3 4 8) (9 11 22) (13 99 23). You must read each partition code sequentially and invert the partition. After that, insert the inverted partition into the back of the sequence of numbers. For example, the first partition code is 1, the front partition (3 4 8) is inverted to (8 4 3) and insert them into the back of the numbers. So the sequence of numbers becomes 9 11 22 13 99 23 8 4 3. Read the next partition code and continue the operation based on the output of the previous the partition-inversion-insertion operation. You must output each list of numbers for each operation.

Note: You must use a linked list; otherwise, no points will be given.

Test Case

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

Listing 3 : Invert a singly linked list

```
Input：
3 4 8 9 11 22 13 99 23
1 2 3 3 1 2

Output：
9 11 22 13 99 23 8 4 3
9 11 22 8 4 3 23 99 13
9 11 22 8 4 3 13 99 23
9 11 22 8 4 3 23 99 13
8 4 3 23 99 13 22 11 9
8 4 3 22 11 9 13 99 23
```

4. Implement a stack by using linked lists

Please finish a program which can push and pop a stack by using a linked list in C. The input includes a sequence of push and pop operations. The output includes two lines of the results: (1) the size of the stack, and (2) the status of the stack from the