Quicksort Partition

*Extracted from:*W00031
Source file name: `quicksort.py`
Time limit: 1

Quicksort is an efficient sorting algorithm based on technique divide and conquer. Below, the Step 1 a divide-and-conquer algorithm:

Step 1: Divide
Choose some pivot element, p, and partition your unsorted array, ar, into three smaller arrays: $left$, $right$, and $equal$, where each element in $left < p$, each element if $right > p$, and each element in $equal = p$.

Given ar and p, (L,C,R), partition ar into $left$, $right$, and $equal$ using the *Divide* instructions above. Then print each element in $left$ followed by each element in $equal$, followed by each element in $right$ on a single line.

The value of pivot $p = L$, is the first value of ar. When $p = R$, is the last value of ar. Finally, $p = C$, is the value of middle ($len//2$) of ar. All elements of each list will be unique.

Input

The first line contains the number of *cases*, the next lines contains information about each case. The first line of each case contains n (the size of ar). The next line contain the value of pivot, p, (L,C,R). The next line contains n space-separated integers describing ar (the unsorted array), the first integer of this line corresponding to ar[0] for this case.

The input must be read from standard input.

Output

For each case, on a single line, print the partitioned numbers (i.e.: the elements in $left$, then the elements in $equal$, and then the elements in $right$). Each integer should be separated by a single space.

The output must be written to standard output.

Sample Input	**Sample Output**
4 4 R 5 3 7 2 5 C 4 5 3 7 2 3 C 3 7 2 6 L 4 8 3 9 2 1	Case 1: 2 5 3 7 Case 2: 2 3 4 5 7 Case 3: 3 2 7 Case 4: 3 2 1 4 8 9

This statement was based by Hackerrank problem: "Quicksort1 Partition" `https://www.hackerrank.com/challenges/quicksort1/problem`.