General Instructions:

- The design mark distribution for each function is given next to it. **Write the design only for those functions.**
- The floating point values in the output should be limited to two decimal places.

---

2. The event coordinators have decided to conduct the final round of the debate event. For this, they collected the scores of all the students participated in the preliminary round from different judges and consolidated the scores. To select the students for the final round, the coordinators need to sort the entire scores in **non-increasing** order. It was observed that in the consolidated list, there were sequences of scores in non-increasing order. The coordinators decided to prepare the final sorted list of scores as follows:

   1. Find the longest sorted sequence (**non-increasing** order) of scores from the consolidated list.
   2. Consider the 3 parts of the consolidated list as,
      – Left sub-array - the scores to the left of the longest sorted sequence (if any).
      – Middle sub-array - the longest sorted sequence.
      – Right sub-array - the scores to the right of the longest sorted sequence (if any).

      Sort the left sub-array (if any) and the right sub-array (if any) separately.
   3. Merge the three sorted sub-arrays using ***3-way merge*** into a single sorted list as described below.
      - Select the largest score from the three sub-arrays and add it to the sorted list, until all the scores are added to the sorted list.

   Write a C program to help the coordinators to prepare the final sorted list. Your program should implement the following functions as per the given function prototypes:

   - $main()$:
     – Read the number of students participated in the preliminary round of debate and their scores, and store the scores in an array A.
     – Repeatedly read a character 'p', 'f', 'l', 'r', 'm', or 't' from the console and call the corresponding functions as described in Input/Output Format section, until character 't' is encountered.
   - $print\_scores(A, i, j)$: Given an array A containing the scores of n students, and two indices i and j such that $0 \le i \le j < n$, print the scores in the array A from index i to index j, separated by space. [0.5 Marks]
   - $longest\_sorted\_sequence(A, n)$: Given the scores of n students in the array A, find the longest sorted sequence (**non-increasing** order) of scores. If there are more than one longest sorted sequence, consider the leftmost one in the array A.

     Note: You may use **two global variables** $start$ and end to store the starting and the ending positions of the longest sorted sequence of scores in the array A. [1 Mark]
   - $sort\_scores(A, n)$: Given the scores of n students in the array A, sort the scores in non-increasing order. [1 Mark]

- $three\_way\_merge(A, d_1, d_2, n)$: Given the scores of n students in the array A, and two indices d_1 and d_2 such that $0 \leq d_1 \leq d_2 <$ n. The left sub-array A[0 .. d_1], the middle sub-array A[d_1+1 .. d_2], and the right sub-array A[d_2+1 .. n-1] are sorted. Merge the three sub-arrays to form a single list sorted in **non-increasing order**, that replaces the array A.
  - In each step, let sc be the score selected for ***3-way merge*** from the three sub-arrays. If there are more than one score eligible for selection, out of them select the score from the right most sub-array as sc.
  - If sc is selected from the left sub-array, print 1; if it is selected from the middle sub-array, print 2; otherwise, print 3.

[1.5 Marks]

Input/Output Format

- The first line contains an integer $n \in [1, 10^3]$ corresponding to the number of students participated in the preliminary round of debate.
- The second line contains n space separated floating point values corresponding to the scores (between 0 and 100, both inclusive) of the n students participated in the preliminary round of debate.

Each of the subsequent lines contains a character from $\{p, f, l, r, m, t\}$. For each of the characters, perform the tasks as specified below.

- Character 'p' : Print the scores of the students stored in the consolidated list, using $print\_scores()$ function.
- Character 'f' :
  1. Find the longest sorted sequence of scores from the consolidated list using $longest\_sorted\_sequence()$ function.
  2. Print the scores in the longest sorted sequence, using $print\_scores()$ function.
- Character 'l' :
  1. Find the longest sorted sequence of scores from the consolidated list using $longest\_sorted\_sequence()$ function.
  2. Sort the left sub-array (if any) using $sort\_scores()$ function.
- Character 'r' :
  1. Find the longest sorted sequence of scores from the consolidated list using $longest\_sorted\_sequence()$ function.
  2. Sort the right sub-array (if any) using $sort\_scores()$ function.
- Character 'm' :
  1. Find the longest sorted sequence of scores from the consolidated list using $longest\_sorted\_sequence()$ function.
  2. Sort the left sub-array (if any) and the right sub-array (if any), using $sort\_scores()$ function.
  3. Merge the three sorted sub-arrays into a single sorted list using $three\_way\_merge()$ function.
- Character 't' : Terminate the program.

Sample Input and Output

Note:

- **The floating point values in the output should be printed with two decimal places (using the format specifier `%0.2f`).**
- The sample input and output are colored ONLY for your better understanding.
- In the input, the colored elements represent the longest sorted sequence.
- In the output, the colored elements represent the sorted sub-array in that step.

Input 1

```
15
2.2 4.23 1.6523 5.3 7.45 5.5 4.9 3.6 2.6 1 9.063 1.45 4.9 2.2 7.451
f
l
p
r
p
m
p
t
```

Output 1

```
7.45 5.50 4.90 3.60 2.60 1.00
5.30 4.23 2.20 1.65 7.45 5.50 4.90 3.60 2.60 1.00 9.06 1.45 4.90 2.20 7.45
5.30 4.23 2.20 1.65 7.45 5.50 4.90 3.60 2.60 1.00 9.06 7.45 4.90 2.20 1.45
3 3 2 2 1 3 2 1 2 2 3 1 1 3 2
9.06 7.45 7.45 5.50 5.30 4.90 4.90 4.23 3.60 2.60 2.20 2.20 1.65 1.45 1.00
```

Input 2

```
15
2.25 1.26 7 9.13 8.25 8.19 7 8.36 1.45 5.9 5.36 4.1 2.3 4.6 3.36
p
f
r
p
l
p
t
```

Output 2

```
2.25 1.26 7.00 9.13 8.25 8.19 7.00 8.36 1.45 5.90 5.36 4.10 2.30 4.60 3.36
9.13 8.25 8.19 7.00
2.25 1.26 7.00 9.13 8.25 8.19 7.00 8.36 5.90 5.36 4.60 4.10 3.36 2.30 1.45
9.13 8.25 8.19 7.00 7.00 2.25 1.26 8.36 5.90 5.36 4.60 4.10 3.36 2.30 1.45
```