QuickSort 1 - Partition

The previous challenges covered Insertion Sort, which is a simple and intuitive sorting algorithm with an average case performance of \( O(n^2) \). In these next few challenges, we’re covering a divide-and-conquer algorithm called QuickSort (also known as Partition Sort).

**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 in \( right \) > \( p \), and each element in \( equal \) = \( p \).

**Challenge**
Given \( ar \) and \( p = ar[0] \), 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. Your output should be space-separated.

**Note:** There is no need to sort the elements in-place; you can create two lists and stitch them together at the end.

**Input Format**
The first line contains \( n \) (the size of \( ar \)).
The second line contains \( n \) space-separated integers describing \( ar \) (the unsorted array). The first integer (corresponding to \( ar[0] \)) is your pivot element, \( p \).

**Constraints**

- \( 1 \leq n \leq 1000 \)
- \(-1000 \leq x \leq 1000, x \in ar\)
- All elements will be unique.
- Multiple answer can exists for the given test case. Print any one of them.

**Output Format**
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.

**Sample Input**

```
5
4 5 3 7 2
```

**Sample Output**

```
3 2 4 5 7
```

**Explanation**

\( ar = [4, 5, 3, 7, 2] \)

*Pivot: \( p = ar[0] = 4 \).*

\( left = \{\}; equal = \{4\}; right = \{\} \).
\[ ar[1] = 5 \geq p, \text{ so it's added to } right. \]
\[ left = \{\}; \ equal = \{4\}; \ right = \{5\} \]

\[ ar[2] = 3 < p, \text{ so it's added to } left. \]
\[ left = \{3\}; \ equal = \{4\}; \ right = \{5\} \]

\[ ar[3] = 7 \geq p, \text{ so it's added to } right. \]
\[ left = \{3\}; \ equal = \{4\}; \ right = \{5, 7\} \]

\[ ar[4] = 2 < p, \text{ so it's added to } left. \]
\[ left = \{3, 2\}; \ equal = \{4\}; \ right = \{5, 7\} \]

We then print the elements of \( left \), followed by \( equal \), followed by \( right \), we get: 3 2 4 5 7.

This example is only one correct answer based on the implementation shown, but it is not the only correct answer (e.g.: another valid solution would be 2 3 4 5 7).