

Quicksort 1 - Partition

Problem Statement

The previous challenges covered *Insertion Sort*, which is a simple and intuitive sorting algorithm. Insertion Sort has a running time of $O(N^2)$ which isn't fast enough for most purposes. Instead, sorting in the real world is done with faster algorithms like Quicksort, which will be covered in the challenges that follow.

The first step of Quicksort is to *partition* an array into two smaller arrays.

Challenge

You're given an array ar and a number p . Partition the array, so that all elements greater than p are to its right, and all elements smaller than p are to its left. p is called the pivot of the partition.

Guideline - In this challenge, you do not need to move around the numbers 'in-place'. This means you can create two lists and combine them at the end.

Input Format

You will be given an array of integers. The number p is the first element in ar .

There are two lines of input:

- n - the number of elements in the array ar
- the n elements of ar (including p at the beginning)

Output Format

Print out the numbers of the partitioned array on one line.

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.

Sample Input

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

Sample Output

```
3 2 4 5 7
```

Explanation

p equals 4. 5 is moved to the right of 4, 2 is moved to the left of 4, and 3 is also moved to the left of 4. The numbers otherwise remain in the same order. Similarly, another valid solution will be $[2, 3, 4, 5, 7]$ and so on.

Task

Complete the method `partition` which takes an array *ar*. The first element in *ar* is the number *p*.