




# Challenge: Implement Partition


The partition function should partition the subarray `array[p..r]` so that all elements in `array[p..q-1]` are less than or equal to `array[q]` (the pivot) and all elements in `array[q+1..r]` are greater than `array[q]`, and it returns the index `q` of where the pivot ends up.

Use the provided `swap()` function for swapping.

 Java

 Python

 C++

 JS

```
1 class Solution {
2     // Swaps two items in an array
3     static void swap(int[] array,
4         int firstIndex,
5         int secondIndex) {
6         int temp = array[firstIndex];
7         array[firstIndex] = array[secondIndex];
8         array[secondIndex] = temp;
9     };
10
11     public static void partition(int[] array,
12         int p, int q, int r) {
13         // Compare array[j] with array[q]
14         // maintaining that:
15         // array[p..q-1] are values less than or equal to array[q]
16         // array[q..j-1] are values greater than array[q]
17         // array[j..r-1] haven't been compared yet
18         // If array[j] > array[q], swap array[j] with array[r]
19         // If array[j] <= array[q], increment j
20         // increment q, and increment j
21         // Once all elements in array[p..r]
22         // have been compared with array[q],
23         // swap array[r] with array[q]
```

