





# Implement Quick Sort

Here we will move on to an intermediate sorting algorithm: quick sort. Quick sort is an efficient, recursive divide-and-conquer approach to sorting an array. In this method, a pivot value is chosen in the original array. The array is then partitioned into two subarrays of values less than and greater than the pivot value. We then combine the result of recursively calling the quick sort algorithm on both sub-arrays. This continues until the base case of an empty or single-item array is reached, which we return. The unwinding of the recursive calls return us the sorted array.

Quick sort is a very efficient sorting method, providing  $O(n\log(n))$  performance on average. It is also relatively easy to implement. These attributes make it a popular and useful sorting method.

**Instructions:** Write a function `quickSort` which takes an array of integers as input and returns an array of these integers in sorted order from least to greatest. While the choice of the pivot value is important, any pivot will do for our purposes here. For simplicity, the first or last element could be used.

	<code>quickSort</code> should be a function.
	<code>quickSort</code> should return a sorted array (least to greatest).
	<code>quickSort([1,4,2,8,345,123,43,32,5643,63,123,43,2,55,1,234,92])</code> should return an array that is unchanged except for order.
	<code>quickSort</code> should not use the built-in <code>.sort()</code> method.