Solution 1

```python
# Copyright © 2020 AlgoExpert, LLC. All rights reserved.

# Best: O(nlog(n)) time | O(log(n)) space
# Average: O(nlog(n)) time | O(log(n)) space
# Worst: O(n^2) time | O(log(n)) space
def quickSort(array):
    quickSortHelper(array, 0, len(array) - 1)
    return array


def quickSortHelper(array, startIdx, endIdx):
    if startIdx >= endIdx:
        return
    pivotIdx = startIdx
    leftIdx = startIdx + 1
    rightIdx = endIdx
    while rightIdx >= leftIdx:
        if array[leftIdx] > array[pivotIdx] and array[rightIdx] < array[pivotIdx]:
            swap(leftIdx, rightIdx, array)
        if array[leftIdx] <= array[pivotIdx]:
            leftIdx += 1
        if array[rightIdx] >= array[pivotIdx]:
            rightIdx -= 1
    swap(pivotIdx, rightIdx, array)
    leftSubarrayIsSmaller = rightIdx - 1 - startIdx < endIdx - (rightIdx + 1)
    if leftSubarrayIsSmaller:
        quickSortHelper(array, startIdx, rightIdx - 1)
        quickSortHelper(array, rightIdx + 1, endIdx)
    else:
        quickSortHelper(array, rightIdx + 1, endIdx)
        quickSortHelper(array, startIdx, rightIdx - 1)


def swap(i, j, array):
    array[i], array[j] = array[j], array[i]

```