

Our Solution(s)

Run Code

Your Solutions

Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 public class Program {
4     // Best: O(nlog(n)) time | O(log(n)) space
5     // Average: O(nlog(n)) time | O(log(n)) space
6     // Worst: O(n^2) time | O(log(n)) space
7     public static int[] QuickSort(int[] array) {
8         QuickSort(array, 0, array.Length - 1);
9         return array;
10    }
11
12    public static void QuickSort(int[] array, int startIdx, int endIdx)
13        if (startIdx >= endIdx) {
14            return;
15        }
16        int pivotIdx = startIdx;
17        int leftIdx = startIdx + 1;
18        int rightIdx = endIdx;
19        while (rightIdx >= leftIdx) {
20            if (array[leftIdx] > array[pivotIdx] && array[rightIdx] < array[pivotIdx]) {
21                swap(leftIdx, rightIdx, array);
22            }
23            if (array[leftIdx] <= array[pivotIdx]) {
24                leftIdx += 1;
25            }
26            if (array[rightIdx] >= array[pivotIdx]) {
27                rightIdx -= 1;
28            }
29        }
30        swap(pivotIdx, rightIdx, array);
31        bool leftSubarrayIsSmaller = rightIdx - 1 - startIdx < endIdx - (rightIdx + 1);
32        if (leftSubarrayIsSmaller) {
33            QuickSort(array, startIdx, rightIdx - 1);
34        }
35        QuickSort(array, rightIdx + 1, endIdx);
36    }
37
38    private static void swap(int i, int j, int[] array) {
39        int temp = array[i];
40        array[i] = array[j];
41        array[j] = temp;
42    }
43 }
```

Solution 1   Solution 2   Solution 3

```
1 public class Program {
2     public static int[] QuickSort(int[] array) {
3         // Write your code here.
4         return null;
5     }
6 }
7
```

Our Tests

Custom Output

Submit Code

```
1 public class Program {
2     // Best: O(nlog(n)) time | O(log(n)) space
3     // Average: O(nlog(n)) time | O(log(n)) space
4     // Worst: O(n^2) time | O(log(n)) space
5     public static int[] QuickSort(int[] array) {
6         QuickSort(array, 0, array.Length - 1);
7         return array;
8     }
9
10    public static void QuickSort(int[] array, int startIdx, int endIdx)
11        if (startIdx >= endIdx) {
12            return;
13        }
14        int pivotIdx = startIdx;
15        int leftIdx = startIdx + 1;
16        int rightIdx = endIdx;
17        while (rightIdx >= leftIdx) {
18            if (array[leftIdx] > array[pivotIdx] && array[rightIdx] < array[pivotIdx]) {
19                swap(leftIdx, rightIdx, array);
20            }
21            if (array[leftIdx] <= array[pivotIdx]) {
22                leftIdx += 1;
23            }
24            if (array[rightIdx] >= array[pivotIdx]) {
25                rightIdx -= 1;
26            }
27        }
28        swap(pivotIdx, rightIdx, array);
29        bool leftSubarrayIsSmaller = rightIdx - 1 - startIdx < endIdx - (rightIdx + 1);
30        if (leftSubarrayIsSmaller) {
31            QuickSort(array, startIdx, rightIdx - 1);
32        }
33        QuickSort(array, rightIdx + 1, endIdx);
34    }
35
36    private static void swap(int i, int j, int[] array) {
37        int temp = array[i];
38        array[i] = array[j];
39        array[j] = temp;
40    }
41 }
```

```
1 public class Program {
2     public static int[] QuickSort(int[] array) {
3         // Write your code here.
4         return null;
5     }
6 }
7
```

```
10 (Test)
11 assert test_helper()
12 test_helper = (0, 0)
13 test_helper = (0, 0)
14 assert test_helper == (0, 0)
15
16 (Test)
17 assert test_helper()
18 test_helper = (0, 0)
19 test_helper = (0, 0)
20 assert test_helper == (0, 0)
21
22
```

Run or submit code when you're ready.