```
// Copyright © 2020 AlgoExpert, LLC. All rights reserved.

// Best: O(nlog(n)) time | O(nlog(n)) space
// Average: O(nlog(n)) time | O(nlog(n)) space
// Worst: O(nlog(n)) time | O(nlog(n)) space
function mergeSort(array) {
  if (array.length <= 1) return array;
  const middleIdx = Math.floor(array.length / 2);
  const leftHalf = array.slice(0, middleIdx);
  const rightHalf = array.slice(middleIdx);
  return mergeSortedArrays(mergeSort(leftHalf), mergeSort(rightHalf));
}

function mergeSortedArrays(leftHalf, rightHalf) {
  const sortedArray = new Array(leftHalf.length + rightHalf.length);
  let k = 0;
  let i = 0;
  let j = 0;
  while (i < leftHalf.length && j < rightHalf.length) {
    if (leftHalf[i] <= rightHalf[j]) {
      sortedArray[k++] = leftHalf[i++];
    } else {
      sortedArray[k++] = rightHalf[j++];
    }
  }
  while (i < leftHalf.length) {
    sortedArray[k++] = leftHalf[i++];
  }
  while (j < rightHalf.length) {
    sortedArray[k++] = rightHalf[j++];
  }
  return sortedArray;
}

exports.mergeSort = mergeSort;

```