Solution 1   Solution 2

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

using System.Linq;

public class Program {
  // 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
  public static int[] MergeSort(int[] array) {
    if (array.Length <= 1) {
      return array;
    }
    int middleIdx = array.Length / 2;
    int[] leftHalf = array.Take(middleIdx).ToArray();
    int[] rightHalf = array.Skip(middleIdx).ToArray();
    return mergeSortedArrays(MergeSort(leftHalf), MergeSort(rightHalf));
  }

  public static int[] mergeSortedArrays(int[] leftHalf, int[] rightHalf) {
    int[] sortedArray = new int[leftHalf.Length + rightHalf.Length];
    int k = 0;
    int i = 0;
    int 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;
  }
}
```