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

import java.util.Arrays;

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 = Arrays.copyOfRange(array, 0, middleIdx);
    int[] rightHalf = Arrays.copyOfRange(array, middleIdx, array.length);
    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;
  }
}

```