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

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
    func mergeSort(_ array: inout [Int]) -> [Int] {
        if array.count <= 1 {
            return array
        }

        let middleIndex = Int(Double(array.count / 2).rounded(.down))
        var leftHalf = Array(array[0 ..< middleIndex])
        var rightHalf = Array(array[middleIndex ..< array.count])

        return mergeSortedArrays(mergeSort(&leftHalf), mergeSort(&rightHalf))
    }

    func mergeSortedArrays(_ leftHalf: [Int], _ rightHalf: [Int]) -> [Int] {
        var sortedArray = Array(repeating: 0, count: leftHalf.count + rightHalf.count)

        var k = 0, i = 0, j = 0

        while i < leftHalf.count, j < rightHalf.count {
            if leftHalf[i] <= rightHalf[j] {
                sortedArray[k] = leftHalf[i]
                i += 1
            } else {
                sortedArray[k] = rightHalf[j]
                j += 1
            }
            k += 1
        }

        while i < leftHalf.count {
            sortedArray[k] = leftHalf[i]
            i += 1
            k += 1
        }

        while j < rightHalf.count {
            sortedArray[k] = rightHalf[j]
            j += 1
            k += 1
        }

        return sortedArray
    }
}

```